

①

文書処理方法及び文書処理装置

特開平 6 - 5 2 1 6 1

(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号

特開平6-52161

(43)公開日 平成 6 年(1994) 2 月 25 日

(51)Int.Cl.⁵

G 0 6 F 15/20

識別記号

5 5 0 E 9288-5L

庁内整理番号

F I

技術表示箇所

審査請求 未請求 請求項の数10(全 32 頁)

(21)出願番号 特願平4-206722

(22)出願日 平成 4 年(1992) 8 月 3 日

(71)出願人 000005496

富士ゼロックス株式会社

東京都港区赤坂三丁目 3 番 5 号

(72)発明者 中津山 恒

神奈川県横浜市保土ヶ谷区神戸町134番地

横浜ビジネスパーク イーストタワー

富士ゼロックス株式会社内

(72)発明者 楠本 浩二

神奈川県横浜市保土ヶ谷区神戸町134番地

横浜ビジネスパーク イーストタワー

富士ゼロックス株式会社内

(74)代理人 弁理士 木村 高久

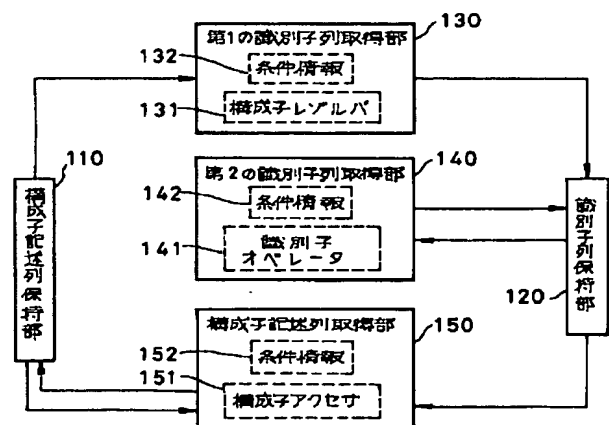
最終頁に続く

(54)【発明の名称】 文書処理方法及び文書処理装置

(57)【要約】

【目的】他の文書処理機能と組合わせて、文書或いは文書部品の検索、挿入、合成、更新を正確に処理することのできる文書処理方法及び文書処理装置を提供する。

【構成】構成子記述列保持部 110 には、予め記述された文書の内容である構成子記述列群が格納されている。第 1 の識別子列取得部 130 は、指定された第 1 の条件に適合する構成子に対応する識別子列を、構成子記述列保持部 110 内の構成子記述列群から取得する。この取得結果である識別子列は、識別子列保持部 120 に保持される。第 2 の識別子列取得部 140 は、識別子列保持部 120 に保持されている識別子列から、指定された第 2 の条件に適合する識別子列を取得する。また取得した識別子列を、既に保持されている他の識別子列と区別できるように識別子列保持部 120 に格納する。構成子記述列取得部 150 は、識別子列保持部 120 に保持されている第 2 の識別子列取得部 140 により得られた識別子列に対応する構成子記述列を、構成子記述列保持部 110 内の指定された構成子記述列から取得する。



【特許請求の範囲】

【請求項 1】 グラフ理論におけるグラフによって表現される文書に対する所定の処理を行う文書処理方法において、

予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている 1 つ以上の構成子を有する構成子記述列から、指定された第 1 の条件に適合する構成子に対応する識別子列を取得するステップと、

前記識別子列から、指定された第 2 の条件に適合する識別子列を取得するステップと、

前記第 2 の条件に適合する識別子列に基づいて、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に対する所定の処理を実行するステップとを含むことを特徴とする文書処理方法。

【請求項 2】 前記所定の処理を実行するステップは、前記第 2 の条件に適合する識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列から取得するステップ、

或いは前記第 2 の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入するステップ、

或いは前記第 2 の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って変更するステップ、

或いは前記第 2 の条件に適合する識別子列を検査し、この検査結果が指定された第 3 の条件に適合する場合のみに、当該識別子列に対応する構成子群を有する前記予め記述された単数又は複数の文書の文書名を出力するステップのうち、いずれかのステップ又は複数のステップを含むことを特徴とする請求項 1 記載の文書処理方法。

【請求項 3】 グラフ理論におけるグラフによって表現される文書の部品を検索する文書処理装置において、

予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている 1 つ以上の構成子を有する構成子記述列から、指定された第 1 の条件に適合する構成子に対応する識別子列を取得する第 1 の識別子列取得手段と、

前記第 1 の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、

前記識別子列保持手段に保持されている識別子列から、指定された第 2 の条件に適合する識別子列を取得する第 2 の識別子列取得手段と、

前記第 2 の識別子列取得手段により得られた識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列から取得する構成子記述列取得手段とを具え

たことを特徴とする文書処理装置。

【請求項 4】 グラフ理論におけるグラフによって表現される文書の部品を所定の文書に挿入する文書処理装置において、

05 予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている 1 つ以上の構成子を有する構成子記述列から、指定された第 1 の条件に適合する構成子に対応する識別子列を取得する第 1 の識別子列取得手段と、

10 前記第 1 の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、

前記識別子列保持手段に保持されている識別子列から、指定された第 2 の条件に適合する識別子列を取得する第 2 の識別子列取得手段と、

15 前記第 2 の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入する構成子記述列挿入手段とを具えたことを特徴とする文書処理装置。

【請求項 5】 グラフ理論におけるグラフによって表現される文書の部品の属性値を更新する文書処理装置において、

25 予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている 1 つ以上の構成子を有する構成子記述列から、指定された第 1 の条件に適合する構成子に対応する識別子列を取得する第 1 の識別子列取得手段と、

30 前記第 1 の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、

前記識別子列保持手段に保持されている識別子列から、指定された第 2 の条件に適合する識別子列を取得する第 2 の識別子列取得手段と、

35 前記第 2 の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段とを具えたことを特徴とする文書処理装置。

【請求項 6】 グラフ理論におけるグラフによって表現される文書を検索する文書処理装置において、

40 予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている 1 つ以上の構成子を有する構成子記述列から、指定された第 1 の条件に適合する構成子に対応する識別子列を取得する第 1 の識別子列取得手段と、

45 前記第 1 の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、

前記識別子列保持手段に保持されている識別子列から、指定された第 2 の条件を満たす識別子列を取得する第 2 の識別子列取得手段と、

50 前記第 2 の識別子列取得手段により得られた識別子列を

検査し、この結果が指定された第 3 の条件に適合する場合は、当該識別子列に対応する構成子群を有する前記予め記述された単数又は複数の文書の文書名を出力する文書名出力手段とを具えたことを特徴とする文書処理装置。

【請求項 7】前記構成子記述列取得手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入する構成子記述列挿入手段を更に具えたことを特徴とする請求項 3 記載の文書処理装置。

【請求項 8】前記構成子記述列取得手段により得られた構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段を更に具えたことを特徴とする請求項 3 記載の文書処理装置。

【請求項 9】グラフ理論におけるグラフによって表現される文書に対する処理を行う文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている 1 つ以上の構成子を有する構成子記述列から、指定された第 1 の条件に適合する構成子に対応する識別子列を取得する第 1 の識別子列取得手段と、前記第 1 の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、前記識別子列保持手段に保持されている識別子列から、指定された第 2 の条件に適合する識別子列を取得する第 2 の識別子列取得手段と、前記第 2 の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段と、前記構成子記述列変更手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する構成子記述列挿入手段とを具えたことを特徴とする文書処理装置。

【請求項 10】グラフ理論におけるグラフによって表現される文書に対する処理を行う文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている 1 つ以上の構成子を有する構成子記述列から、指定された第 1 の条件に適合する構成子に対応する識別子列を取得する第 1 の識別子列取得手段と、前記第 1 の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、前記識別子列保持手段に保持されている識別子列から、指定された第 2 の条件に適合する識別子列を取得する第 2 の識別子列取得手段と、前記予め記述された単数又は複数の文書の構成子記述列から、前記第 2 の識別子列取得手段により得られた識別子列に対応する構成子記述列を取得する構成子記述列取

得手段と、

前記構成子記述列取得手段により得られた構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段と、

05 前記構成子記述列変更手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する構成子記述列挿入手段とを具えたことを特徴とする文書処理装置。

10 【発明の詳細な説明】

【0001】

【産業上の利用分野】この発明は、グラフ理論におけるグラフによって表現される文書を処理する文書処理方法及び文書処理装置に関する。

15 【0002】

【従来の技術野】電子文書の論理構造をグラフ（木や一般のネットワーク）によって表すことは従来から広く行われている。例えば国際規格 ODA（ISO/IS 8613, Information Processing-Text and Office System-Office Document Architecture (ODA) and Interchange Format (1988)）では、文書の論理構造をノード（オブジェクト）によって表現している。またハイパーテキストでは、ノードとリンクとで構成するネットワークが文書である。

20 【0003】論理構造を利用して、記憶装置に記憶された文書を対象として、文書や文書部品の検索、挿入、合成、更新を行う文書処理装置が実現されている。このような文書処理装置を用いて、文書や文書部品の検索、挿入、合成、更新を行うには、通常、問合せ言語を用いて条件や動作を指定する。

30 【0004】文書処理装置においては、何等かの手段によって、文書中のノードのうち処理の対象となるものを特定する必要がある。

35 【0005】この処理の対象となるものを特定する文書処理装置としては、本願出願人によって出願された特願平 2 - 2 3 5 7 4 0 号（発明の名称：電子文書生成方式）、特願平 4 - 1 5 4 6 3 号（発明の名称：文書データベース装置）に開示されたものがある。

40 【0006】特願平 2 - 2 3 5 7 4 0 号に開示されたものにおいては、図 18 に示す様な論文の論理構造を有する構造化文書である入力文書群から出力文書を得るために、以下に述べる様な LISP 言語に似た構造を有する手続き的な操作言語によって記述されたプログラムを実行するようにしている。

45 【0007】文書プログラム識別子：論文抄録作成
(SetID “論文抄録作成”
(MakeRoot (MakeNewNode “論文抄録”))
(AddChild 論文抄録 (MakeNewNode “文献 1”))
(AddChild 論文抄録 (MakeNewNode “文献 2”))
(AddChild 論文抄録 (MakeNewNode “文献 3”))
(AddSubTree 文献 1 表題@論文 A)

(AddSubTree 文献1 著者名@論文A)
(AddSubTree 文献1 所属@論文A)
(AddSubTree 文献1 要約@論文A)
(AddSubTree 文献2 表題@論文B)
(AddSubTree 文献2 著者名@論文B)
(AddSubTree 文献2 所属@論文B)
(AddSubTree 文献2 要約@論文B)
(AddSubTree 文献3 表題@論文C)
(AddSubTree 文献3 著者名@論文C)
(AddSubTree 文献3 所属@論文C)
(AddSubTree 文献3 要約@論文C)
(ChangeText “えつくす研究所” “Our laborator
y”)

(SetAttribute “編集者” “Auto”)

(SetAttribute “編集日” (Date))

図18に示す様な入力文書に対し、このようなプログラムを実行することにより、図19に示す様な出力文書が得られる。

【0008】一方、特願平4-15463号に開示されたものにおいては、図20に示すような論文の論理構造を有する構造化文書からの文書部品の抽出のために、以下に述べるようなSQL言語に似た問合せ言語によって記述されたプログラムを実行するようにしている。

【0009】

```
Project
Project
  タイトル
  著者名
Project
  段落タイトル
Collapse
From 段落
From *
From /データベース関連研究/論文
```

図20に示す様な文書に対し、このようなプログラムを実行することにより、図21に示す様な結果が得られる。この文書による検索結果は文書あるいは文書部品の並びである。

【0010】これらの公報に開示されたものでは、ノードの属性値についての条件を指定することによって、処理の対象となるノードを特定する方法をとっている。この方法では、例えば、「著者」というタグが付随している」という条件の指定を行って、処理の対象となるノードを特定する。

【0011】文書部品の検索は、上記の様な指定によって特定されたノードを、文書中から抜き出すことである。また文書部品の属性更新は、特定されたノードに属性を追加、削除、変更することである。さらに文書部品の挿入は、特定されたノードからの相対位置（例えば直後）を指定して、指定されたノードを挿入することであ

る。

【0012】

【発明が解決しようとする課題】しかしながら、上記従来の各公報に開示されたものでは、文書あるいは文書部品の操作は、全て操作言語で記述されている必要がある。従って、文書部品の抽出、構造の変更、属性の設定、テキストの変更といった一連の処理は、一つの処理系で一括して行われる。

【0013】これらの処理方式においては、ひとつの処理を行う際に、操作言語の処理系である文書処理機能（装置）と他の文書処理機能（装置）とを連携させ、きめ細かな処理を行う事は極めて難しい。

【0014】以上の問題点を具体例を用いて説明する。ノードがテキストで表現されている場合は、汎用のテキスト処理機能（装置）も文書処理機能（装置）として機能し得る。従って、この汎用テキスト処理機能（装置）と、上述した文書処理機能（装置）（上記各公報に開示されたもの）とを組合せて、

（1）文書処理機能によって、操作対象となるノード列を取り出す。

（2）テキスト処理機能によって、ノード列に含まれる文字列の挿入、置換を行う。

（3）文書処理機能によって、上記（2）の処理で得られたノード列を他のノード列に挿入する。

といった処理を行うことが考えられる。

【0015】ところが、汎用テキスト処理機能と、問合せ言語の処理系である従来の文書処理機能とを組合せて使用することは容易ではない。そのため上記（1）～

（3）の処理を分けて実現することができず、以下の問題が生じる。

【0016】テキスト処理機能における操作対象は、上記（1）の処理で取り出された中で更に限定されたノード列のみであるが、この処理が全て終わったときに得られるノード列全体に対して、上記（2）の処理の操作を行うことになってしまう。このため、本来は変更すべきでないノード列も変更されてしまうことがある。

【0017】また上記（1）の処理によって文書から操作対象となるノード列を取り出し、上記（2）の処理が終了した後、元の文書に戻そうとしても、各々のノード列がその文書の中で属していた部分を特定することができない。何故なら、ノードの属性値についての条件を指定することによってノード列を特定したので、ノード列が元の文書でどこに属していたか正確にはわからない。このため上記（3）の処理を行うことができない。

【0018】このように従来の文書処理機能では、他の文書処理機能と組合せてきめ細かな処理を行うことは出来なかった。

【0019】この発明は、他の文書処理機能と組み合わせ、文書あるいは文書部品の検索、挿入、合成、更新を正確に処理することのできる文書処理方法及び文書処

理装置を提供することを目的とする。

【0020】

【課題を解決するための手段】第1の発明は、グラフ理論におけるグラフによって表現される文書に対する所定の処理を行う文書処理方法において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得するステップと、前記識別子列から、指定された第2の条件に適合する識別子列を取得するステップと、前記第2の条件に適合する識別子列に基づいて、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に対する所定の処理を実行するステップとを含むことを特徴とする。

【0021】第2の発明は、第1の発明において、前記所定の処理を実行するステップは、前記第2の条件に適合する識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列から取得するステップ、前記第2の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入するステップ、前記第2の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って変更するステップ、前記第2の条件に適合する識別子列を検査し、この検査結果が指定された第3の条件に適合する場合のみに、当該識別子列に対応する構成子群を有する前記予め記述された単数又は複数の文書の文書名を出力するステップのうち、いずれかのステップ又は複数のステップを含むことを特徴とする。

【0022】第3の発明は、グラフ理論におけるグラフによって表現される文書の部品を検索する文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得する第1の識別子列取得手段と、該第1の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第2の条件に適合する識別子列を取得する第2の識別子列取得手段と、該第2の識別子列取得手段により得られた識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列から取得する構成子記述列取得手段とを具えている。

【0023】第4の発明は、グラフ理論におけるグラフによって表現される文書の部品を所定の文書に挿入する文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識

別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得する第1の識別子列取得手段と、該第1の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第2の条件に適合する識別子列を取得する第2の識別子列取得手段と、該第2の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入する構成子記述列挿入手段とを具えている。

【0024】第5の発明は、グラフ理論におけるグラフによって表現される文書の部品の属性値を更新する文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得する第1の識別子列取得手段と、該第1の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第2の条件に適合する識別子列を取得する第2の識別子列取得手段と、該第2の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段とを具えている。

【0025】第6の発明では、グラフ理論におけるグラフによって表現される文書を検索する文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得する第1の識別子列取得手段と、該第1の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第2の条件を満たす識別子列を取得する第2の識別子列取得手段と、該第2の識別子列取得手段により得られた識別子列を検査し、この結果が指定された第3の条件に適合する場合は、当該識別子列に対応する構成子群を有する前記予め記述された単数又は複数の文書の文書名を出力する文書名出力手段とを具えている。

【0026】第7の発明は、第3の発明において、前記構成子記述列取得手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入する構成子記述列挿入手段を更に具えたことを特徴とする。

【0027】第8の発明は、第3の発明において、前記構成子記述列取得手段により得られた構成子記述列を、

指定された変更指示情報に従って変更する構成子記述列変更手段を更に具えたこと特徴とする。

【0028】第9の発明は、グラフ理論におけるグラフによって表現される文書に対する処理を行う文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得する第1の識別子列取得手段と、該第1の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第2の条件に適合する識別子列を取得する第2の識別子列取得手段と、該第2の識別子列取得手段により得られた識別子列に対応する構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段と、該構成子記述列変更手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する構成子記述列挿入手段とを具えている。

【0029】第10の発明は、グラフ理論におけるグラフによって表現される文書に対する処理を行う文書処理装置において、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得する第1の識別子列取得手段と、該第1の識別子列取得手段により得られた識別子列を保持する識別子列保持手段と、該識別子列保持手段に保持されている識別子列から、指定された第2の条件に適合する識別子列を取得する第2の識別子列取得手段と、前記予め記述された単数又は複数の文書の構成子記述列から、前記第2の識別子列取得手段により得られた識別子列に対応する構成子記述列を取得する構成子記述列取得手段と、該構成子記述列取得手段により得られた構成子記述列を、指定された変更指示情報に従って変更する構成子記述列変更手段と、該構成子記述列変更手段により得られた前記構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する構成子記述列挿入手段とを具えている。

【0030】

【作用】第1の発明では、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ、識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得し、この識別子列から、指定された第2の条件に適合する識別子列を取得し、この第2の条件に適合する識別子列に基づいて、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部

品に対応する構成子記述列に対する所定の処理を実行する。

【0031】第2の発明では、第1の発明において、前記所定の処理をするに際し、前記第2の条件に適合する識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列から取得する処理、前記第2の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応する構成子記述列に挿入する処理、前記第2の条件に適合する識別子列に対応する構成子記述列を、指定された条件に従って変更する処理、前記第2の条件に適合する識別子列を検査し、この検査結果が指定された第3の条件に適合する場合のみに、前記予め記述された単数又は複数の文書の文書名を出力する処理のうち、いずれかの処理又は複数の処理を実行する。

【0032】第3の発明では、第1の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得し、更にこの識別子列を識別子列保持手段に格納する。第2の識別子列取得手段が、該識別子列保持手段に保持されている識別子列中から、指定された第2の条件に適合する識別子列を取得し、構成子記述列取得手段が、この識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列から取得する。従って、指定された文書から条件に適合する文書部品を検索することができる。

【0033】第4の発明では、第1の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得し、更にこの識別子列を識別子列保持手段に格納する。第2の識別子列取得手段が、該識別子列保持手段に保持されている識別子列中から、指定された第2の条件に適合する識別子列を取得し、構成子記述列挿入手段が、この識別子列に対応する構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する。従って、ある文書中の文書部品を抽出し、これを他の文書中に挿入することができる。

【0034】第5の発明では、第1の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取

得し、更にこの識別子列を識別子列保持手段に格納する。第2の識別子列取得手段が、該識別子列保持手段に保持されている識別子列中から、指定された第2の条件に適合する識別子列を取得し、構成子記述列変更手段が、この識別子列に対応する構成子記述列を、指定された変更指示情報に従って変更する。従って、指定された文書中の文書部品の属性を指定された属性値に変更することができる。

【0035】第6の発明では、第1の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得し、更にこの識別子列を識別子列保持手段に格納する。第2の識別子列取得手段が、該識別子列保持手段に保持されている識別子列中から、指定された第2の条件に適合する識別子列を取得し、文書名出力手段が、この識別子列を検査し、この結果が指定された第3の条件に適合する場合は、当該識別子列に対応する構成子群を有する前記予め記述された単数又は複数の文書の文書名を出力する。従って、指定された条件に適合する文書の文書名のみを得ることができる。

【0036】第7の発明では、第1の発明において、構成子記述列挿入手段が、前記構成子記述列取得手段により得られた構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する。従って、指定された文書中から指定された条件に適合する文書部品を検索し、この文書部品を他の文書中に挿入することができる。

【0037】第8の発明では、第1の発明において、構成子記述列変更手段が、前記構成子記述列取得手段により得られた構成子記述列を、指定された変更指示情報に従って変更する。従って、指定された文書中から指定された条件に適合する文書部品を検索し、この文書部品の属性を指定された属性値に変更することができる。

【0038】第9の発明では、第1の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得し、更にこの識別子列を識別子列保持手段に格納する。第2の識別子列取得手段が、該識別子列保持手段に保持されている識別子列中から、指定された第2の条件に適合する識別子列を取得し、構成子記述列変更手段が、この識別子列に対応する構成子記述列を指定された変更指示情報に従って変更し、更に構成子記述列挿入手段が、この変更処理後の構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入

る。従って、指定された文書中から指定された条件に適合する文書部品を検索し、該文書部品の属性を指定された属性値に変更することができる。

【0039】第10の発明では、第1の識別子列取得手段が、予め記述された単数又は複数の文書或いは単数又は複数の文書部品に対応し、且つ識別子が付与されている1つ以上の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得し、更に、この識別子列を識別子列保持手段に格納する。第2の識別子列取得手段が、該識別子列保持手段に保持されている識別子列から、指定された第2の条件に適合する識別子列を取得し、構成子記述列取得手段が、この識別子列に対応する構成子記述列を、前記予め記述された単数又は複数の文書の構成子記述列から取得し、構成子記述列変更手段が、この構成子記述列を指定された変更指示情報に従って変更し、更に構成子記述列挿入手段が、この変更処理後の構成子記述列を、指定された条件に従って、前記予め記述された単数又は複数の文書或いは単数又は複数の文書部品の構成子記述列に挿入する。従って、指定された文書中から指定された条件に適合する文書部品を検索し、この文書部品の属性を指定された属性値に変更し、更に、この文書部品を指定された文書中に挿入することができる。

【0040】

【実施例】以下、本発明の第1～第5の実施例を添付図面を参照して説明する。

【0041】第1の実施例を図1乃至図11を参照して説明する。

【0042】図1は本発明に係る文書処理装置の第1の実施例を機能ブロックで示したものである。

【0043】同図において、文書処理装置は、構成子記述列保持部110と、識別子列保持部120と、第1の識別子列取得部130と、第2の識別子列取得部140と、構成子記述列取得部150とを有して構成されている。

【0044】構成子記述列保持部110には構造化文書である電子文書が単数又は複数格納されている。この文書の論理構造はグラフ理論におけるグラフによって表現することができるので、この実施例においては、グラフのノードを構成子と定義し、また構成子に対応する記述を構成子記述と定義し、更に構成子を複数並べたものを構成子記述列と定義すると、文書全体は、グラフの全てのノードをそれぞれ表す構成子記述列によって表現される。テキストや図形など、文書の内容を持つ構成子を持つ内容部と呼ぶ。また、文書の部品（例えば、章、見出し、図など）も、それを構成するノードを表す構成子記述列によって表現される。

【0045】ここで、論理構造の構成子記述列の例を以下に示す。

```

logical-object {
  object-type document-logical-root ,
  description-body {
    object-identifier "3",
    suordinates "2",
    user-visible-name "文書論理根" } } ,
logical-object {
  object-type composite-logical ,
  description-body {
    object-identifier "3 0",
    suordinates "2",
    user-visible-name "第1章" } } ,
logical-object {
  object-type basic-logical ,
  description-body {
    object-identifier "3 0 0",
    user-visible-name "1.1 節" } } ,
logical-object {
  object-type basic-logical ,
  description-body {
    object-identifier "3 0 1",
    user-visible-name "1.2 節" } } ,
logical-object {
  object-type basic-logical ,
  description-body {
    object-identifier "3 0 2",
    user-visible-name "1.2 節" } }

```

ここで、例えば

```

logical-object {
  object-type document-logical-root ,
  description-body {
    object-identifier "3",
    suordinates "2",
    user-visible-name "文書論理根" } } ,

```

で記述される様に「logical-object { }」が構成子であり、「object-identifier "3"」が識別子である。

【0046】なお上記の例では、5つの構成子で意味のある1つの構成子記述列が表現されている。また上記の構成子記述列から、ある条件に従って、例えば任意の2つの構成子が抽出された場合は、その2つの構成子で意味のある1つの構成子記述列が表現されることとなる。

【0047】第1の識別子列取得部130では、構成子レゾルバ131が、予め設定された第1の条件としての条件情報132に基づいて、構成子記述列保持部110内の構成子記述列群から、その条件に適合する構成子に対応する識別子列を取得するものである。なお条件情報132は構成子の検索条件を示す情報である。

【0048】識別子列保持部120は、第1の識別子列取得部130により得られた識別子列を保持する。

【0049】第2の識別子列取得部140では、識別子

オペレータ141が、予め設定された第2の条件としての条件情報142に基づいて、識別子列保持部120に保持されている識別子列から、その条件に適合する識別子列を取得する。なお条件情報142は1つ又は複数の識別子を得るための条件を示す情報である。

【0050】識別子オペレータ141によって取得された識別子列は識別子列保持部120に保持される。なおこのとき、識別子列保持部120には、第1の識別子列取得部130によって取得された識別子列と、第2の識別子列取得部140によって取得された識別子列とは、区別されるように保持される。

【0051】構成子記述列取得部150では、構成子アクセサ151が、構成子記述の取得条件を示す条件情報152に基づいて、構成子記述列保持部110内の構成子記述列群から該当する構成子記述列を取得する。ここでは、構成子アクセサ151が、第2の識別子列取得部140により得られた識別子列を識別子列保持部120から得ると共に、この得た識別子列に対応する構成子記述列を、構成子記述列保持部110内の構成子記述列群から取得するようになっている。こうして取得された構成子記述列は、既に保持されている他の構成子記述列と区別できるように構成子記述列保持部110に保持され、さらに処理操作の対象にもなり得るものである。

【0052】なおこの実施例においては、第1の識別子列取得部130、第2の識別子列取得部140、構成子記述列取得部150は、それぞれの構成要素の機能を遂行させるためのソフトウェア（プログラム）をプロセッサや中央処理装置などの制御手段が実行することにより実現されている。ここでは、前記各構成要素をそれぞれ独立したプログラム（コマンド）として実現し、構成子記述列、識別子列を表すデータ構造を定め、コマンド間で授受するようにしている。勿論、これを、1つのコマンドに2つ以上の手段に相当する機能を持たせるようにしても良い。この場合は、そのコマンドの入出力のインタフェースは各手段を独立なコマンドとした場合のインタフェースに準ずるものとする。なおこの実施例で使

用したコマンドの一例については後述する。

【0053】上記各コマンドは、UNIXの汎用オペレーティングシステムで動作するようになっている。また、各コマンドは単純な機能のみを有するもので、実際に文書処理を行うときには各コマンドが組合わされて使用される。このとき、シェルと呼ばれるコマンドプロセッサ（コマンドインタプリタ）の制御機能を使用して、シェルスクリプトと呼ばれるコマンドプロシジャを作成することにより、高度な処理を実現することができる。あるコマンドの出力を他のコマンド入力とするには、パイプを使用したり、入出力切り替え（リダイレクション）により一時ファイルを生成して使用することができる。

【0054】例えば、パーソナルメディアから1987

年に発行された「落水浩一郎、大木敦男 共訳、UNIX C SHELL フィールドガイド」(原本:「Gail Anderson and Paul Anderson. The UNIX C Shell Field Guide」)の文献を参照することにより、シェルを実現することができる。

【0055】構成子記述列保持部110、識別子列保持部120は、ファイルシステムとパイプ(キュー)を用いて実現している。従ってこれらの構成要素は、格納時に内容と名前を関連付ける機能と、名前で内容をアクセスする機能とを持っており、また内容が一度だけ用いられる場合は、パイプを用いて名前を使わずに一時的に保持する機能を持っている。

【0056】図2は、第1に示した実施例の装置を実現するためのハードウェア構成を示したものであり、例えば、ワークステーションやコンピュータなどの装置のブロック図を示している。

【0057】同図において、装置は、入力装置210、ディスプレイ220、ディスク230、主メモリ240、中央処理装置(以下、CPUという)250とがバス260を介してそれぞれ接続されている。

【0058】入力装置210は、キーボード及びマウスから構成されており、コマンドプロシジャなどの各種のデータの入力や、各種の指令を与えるものである。

【0059】ディスプレイ220は、文書の論理構造、構成子記述列、文書内容、コマンドプロシジャを表示するものである。

【0060】ディスク230には、電子文書(ファイル)つまり文書の論理構造や構成子記述列や文書内容、この実施例の文書処理機能を有するコマンドインタプリタ(プログラム)、コマンドプロシジャ、各種データなどが格納されている。

【0061】主メモリ240には、ディスク230からロードされた電子文書ファイル、コマンドインタプリタ、コマンドプロシジャ、各種データ、及び入力装置210によって入力されたコマンドプロシジャや各種データや指令を記憶する。なお、この主メモリ240には、CPU250がデータに対する操作を行うための作業領域が設けられており、この作業領域中に、キュー(パイプ)が作成される。また主メモリ240に示されている各データやプログラムの配置は、説明用のために配置したものであり、これらは、図示されているデータ構造で配置されるものではない。

【0062】CPU250は、コマンドインタプリタをディスク230から主メモリ240にロードして実行することにより、コマンドプロシジャに対する処理を行う。すなわち、電子文書あるいは文書部品の検索、挿入、合成、更新などの文書処理を行う。またCPU250は、バス260を介してこれに接続された各構成要素を制御する。

【0063】コマンドの説明は、原則的にUNIXのオ

ンラインマニュアルの形式に従っている。従って、大括弧([])で囲まれたものは、それが省略可能であることを示す。ここでは、縦棒(|)と中括弧({ })を導入している。縦棒は、その両方のうちいずれかを選択することを示し、中括弧は、その中の1つが必ず選ばれることを示す。

【0064】次に、図1に示した機能ブロック図の構成要素と図2に示したブロック図との対応関係について説明する。

10 【0065】図1に示した構成子記述列保持部110及び識別子保持部120はディスク230、主メモリ240に対応し、第1の識別子列取得部130、第2の識別子列取得部140、構成子記述列取得部150はCPU250(実際にはCPU250がコマンドインタプリタ

15 を実行して、コマンドを解釈することにより実現される)に対応している。

【0066】なお後述する第2～第5の実施例の装置も図2に示したハードウェア構成で実現されている。

20 【0067】次に、この実施例で使用するコマンドについて説明する。ここでは、以下のようにコマンド群のカテゴリ(範疇)を定義する。

[カテゴリ : 機能説明]
◇構成子レゾルバ : 構成子記述列から条件を満たす識別子列を得る。

25 ◇識別子オペレータ : 識別子列(の列)から条件を満たす識別子列を得る。

◇構成子アクセサ : 識別子列と文書から構成子記述列を得る。

◇部品挿入 : 文書に部品を挿入する。

30 ◇属性更新 : 指定された構成子記述の属性値を更新する。

◇文書オペレータ : 識別子列と文書名から条件を満たす文書名を得る。

35 ◇文書レゾルバ : 文書名の列から条件を満たす文書名の列を得る。

◇文書アクセサ : 文書名から文書の内容を得る。

40 なお、図1に示した構成子レゾルバ131は上記コマンドの構成子レゾルバに相当し、図1に示した識別子オペレータ141は上記コマンドの識別子オペレータに相当し、図1に示した構成子アクセサ151は上記コマンドの構成子アクセサに相当する。

【0068】次に、カテゴリ毎に、カテゴリに属するコマンドの名前、形式、機能、オプションについて説明する。

45 【0069】◇構成子レゾルバ
構成子記述列と条件を入力として、条件を満たす構成子の識別子列を得るコマンドである。条件としては、属性と指定された数値との比較(=、≠、<、>、≤、≥)、属性値と指定された文字列との辞書順比較や照合などがあげられる。このカテゴリに属するコマンドの例

を2つ挙げる。

【0070】 [名前] select-content : 内容部を扱う構成子レゾルバ

【形式】 select-content string [filename]

【機能説明】 filename (ファイル名) で指定された文書ファイルから、stringで与えられた文字列が部分文字列として現れる構成子 (内容部) を求める。大文字・小文字は区別されない (例えばstringとして“a b c”が指定されたとき、“A B C”という文字列とも一致する。)。filenameが与えられなかったときには、標準入力 (例えば端末。以下同様とする) から構成子記述列を読み込む。

【0071】 [名前] select-type : 属性“uobject-type”を扱う構成子レゾルバ

【形式】 select-type [-e] string [filename]

【機能説明】 filename (ファイル名) で指定された文書ファイルから、属性“uobject-type”の値が与えられた条件に適合する構成子の識別子を求め、標準出力 (例えば端末。以下同様とする) に書き出す。filenameが与えられなかったときには、標準入力から構成子記述列を読み込む。-eオプションが指定されない場合は、属性“uobject-type”の値がstringで与えられた文字列と完全に一致するものを求める。なお大文字・小文字は区別されない。

【オプション】 -e : stringが属性“uobject-type”の値の部分文字列として出現する構成子を求める。

【0072】 ◇識別子オペレータ

識別子列と条件を入力として、指定された属性の値が条件を満たすような構成子の識別子を列挙するコマンドである。

【0073】 このカテゴリに属するコマンドの例を、4つ挙げる。

【0074】 [名前] a b o v e

【形式】 above [filename]

【機能説明】 filename (ファイル名) で指定された2つの識別子列を読み、最初の識別子列に含まれる識別子のうち、2番目の識別子列に含まれるいずれかの識別子の祖先になっている識別子を選択し、標準出力に書き出す。filenameが省略された場合には、標準入力から識別子列を読み込む。

【0075】 [名前] b e l o w

【形式】 below [filename]

【機能説明】 filename (ファイル名) で指定された2つの識別子列を読み、最初の識別子列に含まれる識別子のうち、2番目の識別子列に含まれるいずれかの識別子の子孫になっている識別子を選択し、標準出力に書き出す。filenameが省略された場合には、標準入力から識別子列を読み込む。

【0076】 [名前] b o t t o m

【形式】 bottom [filename]

【機能説明】 filename (ファイル名) で指定された識別子列ファイルから、他の識別子の先祖になっていない識別子だけを選び出して標準出力に書き出す。filenameが指定されなければ、標準入力から識別子列を読み込む。

例えば入力として、<3 1> <3 1 2> <3 1 2 3> <3 1 3> <3 3 3>が与えられると (上記例では5つの識別子からなる1つの識別子列)、出力として、<3 1 2 3> <3 1 3> <3 3 3>が得られる (1つの識別子列)。

10 【0077】 [名前] u n i o n

【形式】 union [filename]

【機能説明】 filename (ファイル名) で指定されたファイルに含まれる2つの識別子列から和集合を求め、標準出力する。filenameが指定されなければ、標準入力から識別子列を読み込む。例えば入力として、<3 1> <3 1 2 3> <3 3 3>と<3 1 2> <3 1 2 3> <3 3 3>の2つの識別子列が与えられると (上記例では、3つの識別子列からなる1つの識別子列と、3つの識別子列からなる1つの識別子列とからなる複数の識別子列)、出力として、<3 1> <3 1 2> <3 1 2 3> <3 3 3>が得られる (1つの識別子列)。

【0078】 なおこのunionコマンドの様に、2つの識別子列 (集合) を入力として、集合演算を行う他の例としては、次のようなコマンドがある。

25 (a) intersection : 2つの識別子列 (集合) を入力として共通集合を求める。

(b) difference : 2つの識別子列 (集合) を入力として差集合を求める。

(c) unique : 識別子列を入力として、識別子の集合を求める (重複しないようにする)。

などがある。

【0079】 ◇構成子アクセサ

構成子記述列と識別子と条件を入力として、条件を満たす構成子記述列を得るコマンドである。条件として、識別子オペレータの項で説明した接続情報を用いることができる。条件が与えられなければ、識別子列中の各識別子に対応する構成子の構成子記述を取り出す。

【0080】 このカテゴリのコマンドの例を1つ挙げる。

40 【0081】 [名前] e x t r a c t

【形式】 extract [-s] id-file filename

【機能説明】 id-file (識別子列ファイル) で指定された文書ファイルから、id-file で指定された識別子列に対応する構成子記述を求める。id-file として“-”が与えられたときには、標準入力から識別子を読み込む。filename (ファイル名) として“-”が与えられたときには、標準入力から構成子記述列を読み込む。id-file とfilenameの両方に“-”を与えることはできない。

【オプション】 -s : 指定された識別子を持つ構成子を頂点とする部分木の構成子記述列を取り出す。

【0082】◇部品挿入

挿入箇所を示す識別子と、文書部品ファイル（構成子記述列）と、文書ファイル（構成子記述列）とを入力して、文書ファイルに部品を挿入するコマンドである。部品を挿入したときには、識別子が文書内で一意になるよ

insert [-ec | -yc | -eb | -yb] [-q] {-p id | id-filename}
part-filename doc-filename

【機能説明】構成子記述列を、文書中の指定された挿入位置に挿入して、標準出力に書き出す。構成子記述列と文書ファイルはそれぞれ、part-filename、doc-filenameで指定したファイルから読み込まれる。挿入位置は、識別子ファイルid-filenameの先頭か、-pオプションで指定された識別子を基準に、他のオプションによって決定される。part-filename、doc-filename、id-filenameのうちいずれかに“-”を指定したときは、それを標準入力から読み込む。“-”を複数回指定することはできない。出力中の構成子記述の識別子は、Deweyの10進分類法(Donald E. Knuth, “The Art of Programming”, Addison-Wesley Publishing Company, 1973. 邦訳: 筧捷彦・米田信夫共訳, “基本算法/情報構造”, サイエンス社, 1987年)に従うよう振り直される。

【オプション】

-ec : 指定された識別子を持つ構成子の最初の子供になるように挿入する(eldest child)。
-yc : 指定された識別子を持つ構成子の最後の子供になるように挿入する(youngest child)。
-eb : 指定された識別子を持つ構成子の兄になるように挿入する(elder brother)。
-yb : 指定された識別子を持つ構成子の弟になるように挿入する(younger brother)。
-p id : 構成子記述列を挿入するノードの識別子を指定する。例えば、-p “301”のように指定する。

【0085】◇属性更新

id-doc [-gt | -lt | -ge | -le | -eq | -ne] num {id-file | -} doc-file

【機能説明】id-file（識別子列ファイル）で指定されたファイルに含まれる識別子の数が、与えられた条件を満たすときdoc-file（文書ファイル）を出力する。id-fileの代わりに“-”が与えられたときは、標準入力から識別子列を読み込む。

【オプション】

-gt num : id-fileに含まれる識別子の数がnumより大きいときに真。
-lt num : id-fileに含まれる識別子の数がnumより小さいときに真。
-ge num : id-fileに含まれる識別子の数がnum以上のときに真。
-le num : id-fileに含まれる識別子の数がnum以下のときに真。
-eq num : id-fileに含まれる識別子の数がnumと等

うに、必要に応じて識別子の再割り当てを行う。

【0083】このカテゴリに属するコマンドの例を1つ挙げる。

【0084】[名前] insert

【形式】

構成子記述列と識別子列と変更指示を入力として、指示に従って属性の更新を行うコマンドである。識別子が与えられない場合には、全構成子記述の属性の更新を行う。

【0086】このカテゴリに属するコマンドの例を1つ挙げる。

【0087】[名前] chval

【形式】

chval [-n | -t] old new [id-file | -] doc-file

【機能説明】doc-file（文書ファイル）中の構成子記述列のうち、id-file（識別子列ファイル）の識別子列に対応する構成子の属性値に含まれる文字列oldを文字列newで置換する。id-fileの代わりに、“-”が与えられたときには、標準入力から識別子列を読み込む。id-fileに識別子が全く含まれていないときには、doc-fileのすべての構成子进行操作対象とする。

【オプション】

-n : 属性“user visible name”进行操作対象とする。
-t : 属性“uobject type”进行操作対象とする。

【0088】◇文書オペレータ

識別子列と文書名と条件を入力とし、文書名を得るオペレータである。与えられた条件が満たされればその文書名を出力し、そうでなければ何も出力しない。このカテゴリに属するコマンドの例を1つ挙げる。

【0089】[名前] id-doc

【形式】

id-doc [-gt | -lt | -ge | -le | -eq | -ne] num {id-file | -} doc-file

しいときに真。
-ne num : id-fileに含まれる識別子の数がnumと等しくないときに真。

【0090】◇文書レゾルバ

文書名列と条件を入力として、条件を満たす文書名列を得るコマンドである。このカテゴリに属するコマンドは、指定された属性の値が条件を満たすような文書の名前を列挙する。条件としては、属性と指定された数値の比較(=、≠、<、>、≤、≥)、属性値と指定された文字列の辞書順比較や照合などがあげられる。例えば、文書名、著者、キーワードに特定の文字列を含む文書の列挙、ページ数が指定された値以上（あるいは以下）の文書の列挙、を行うコマンドがある。

【0091】◇文書アクセサ

文書名を引数として文書の内容である構成子記述列を得

るコマンドである。例えば、指定された文書ファイルの内容を標準出力に書き出すコマンドがある。なおこのカテゴリのコマンドの機能は、ファイルシステムに持たせても良い。

【0092】ところで、接続関係に基づく識別子オペレータの例としては、指定された識別子に対応する構成子との距離が指定された値以下である構成子の識別子を列挙すコマンドがある。例えば、図3に示すようなネットワーク構造において、ノード1との距離が1以下のノードは{1, 2}である。なお4つの図形“○”はノード

を表している。

【0093】接続関係に基づく演算を行うコマンドの実現方法について述べる。

【0094】簡単のため、入力となる各構成子記述に対

しては、構成子に文書内で一意な識別子があらかじめ付与されているものとする。

【0095】データ構造としては、識別子を添数とする隣接行列を用いる。識別子は自然数とあるとし、論理構造がなすグラフがストリクトであるとするれば、n個の構成子記述に対して、隣接行列として以下の様なn次の正方行列 $A = a_{ij}$ ($1 \leq i, j \leq n$) を用意することにより、隣接情報を得ることができる。

【0096】ここにストリクトとはループと多重弧が存在しないこと。ここで、ループとはあるノードからそれ自身へのアークであり、多重弧とはあるノードから他の特定のノードに至る複数のアークである。

【0097】

$$a_{ij} = \begin{cases} 1 & \text{アーク}(i, j) \text{が存在する} \\ 0 & \text{アーク}(i, j) \text{が存在しない} \end{cases}$$

例えば、 $A^1 (=A)$ 、 $A^2 (=A \cdot A)$ 、 $\dots A^n$ から、それぞれあるノードから1、2、 \dots 、n回リンクを辿って到達できるノードを求めることができる。

また、 $\sum_{i=1}^n A^i$ から

リンクをn以下の回数だけ辿ったときに到達可能なノードを求めることができる。

【0098】ところで、図3に示すようなネットワークであれば、隣接行列Aは、

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

となる。このとき、

$$A^2 = A \cdot A = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$A^3 = A^2 \cdot A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$A^4 = A^3 \cdot A = 0$$

である。

【0099】Aで定義されるグラフは、DAG (directed acyclic graph) であるので、

$C = \sum_{i=1}^{\infty} A^i$ とすれば、

45

$$C = \sum_{i=1}^3 A^i = \begin{bmatrix} 0 & 1 & 2 & 1 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

となる。

50 【0100】 $C = c_{ij}$ ($0 \leq i, j \leq n$) の成分c

i, j は、ノード i からノード j に至る経路の総数を表す。

【0101】このような接続情報に関する性質を用いて、識別子と条件を入力として、識別子列を出力するコマンドを実現することができる。

【0102】一般に構造化文書の論理構造は木構造で表されるが、木構造は接続に制限を設けた特殊なネットワーク構造であるので、上述した隣接行列を用いて親子関係などの子孫関係を求めることができる。従って、本実施例により、構造化文書もネットワーク構造で表現される文書と全く同様に操作することができる。

【0103】構成子記述がテキストで表現されている場合には、本実施例に組み合わせて、UNIXの標準コマンド `sed`、`awk` といった汎用のテキスト操作ツール（あるいは言語）を用いることができる。

【0104】オブジェクト指向データベースでは、オブジェクトに対して内部的に一意な識別子を割り当てている。従って、オブジェクト指向データベースにおいても、識別子の内部表現に自然数が1対1に対応するようになれば、隣接行列を用いることにより、識別子と条件を入力して識別子列を出力するようなプログラムを作成

$$hd(b, length(a)) = hd(\langle 1212 \rangle, 2) = \langle 12 \rangle = a$$

である。

【0110】2つのアドレスを引数とする関数 `lub` を

$$lub(a, b)$$

$$= hd(a, \max\{i \mid hd(a, i) = hd(b, i)\})$$

`lub(a, b)` はアドレス a, b に対応するオブジェクトの共通の祖先のうち、もっとも若いもののアドレスである。

【0111】この関数は図6に示すような処理手順により、プログラムとして実現することができる。

【0112】第2の識別子列取得部は、アドレス a, b を入力し（ステップ601）、 i を値「1」と定義する（ステップ602）。次に、 $a(i) = b(i)$ の式が成立するか否かを判断し（ステップ603）、成立する場合は、 $a(i)$ を `print`（つまり出力）し（ステップ604）、 i を $i+1$ と再定義する（ステップ605）。その後、 $a(i) = b(i)$ の式が成立するか否かを判断し（ステップ606）、成立する場合は上記ステップ604に戻りこのステップ以降を実行し、不成立

することができる。

【0105】次に、図4に示すような木構造によって表現される文書を処理対象とする場合について説明する。

【0106】ここでは、木構造の各ノードにアドレスを割り当て、アドレスの集合に対する演算を定義し、その演算を行うコマンドの実現方法を説明する。また、構成子アクセスでの処理において、アドレスの集合に対する演算に基づいてアクセスする方法も説明する。

【0107】図5に示すように、図4に示す木構造の各ノードにDeweyの10進分類法によってアドレスを割り当てる。アドレスは数値の並びである。

【0108】このようにして与えられたアドレスから、ただちに接続関係を得ることができる。アドレス `addr` の長さを `length(addr)`、アドレス `addr` の先頭の `len` 桁を `hd(addr, len)` で表すことにすると、アドレス a に対応するオブジェクトがアドレス b に対応するオブジェクトの先祖であるための必要十分条件は、 $a = hd(b, length(a))$ である。

【0109】例えば、 $a = \langle 12 \rangle$ 、 $b = \langle 1212 \rangle$ とすれば、

次のように定義する。

の場合は、リターンする。ステップ603で式が不成立の場合はリターンする。

【0113】次に関数 `lub` を用いて、述語 `anc`、`desc` を定義する。

$$anc(a, b) \Leftrightarrow lub(a, b) = a$$

$$desc(a, b) \Leftrightarrow lub(a, b) = b$$

すなわち、`anc(a, b)` は a が b の祖先であるか等しいとき、かつそのときに限り真、`desc(a, b)` は a が b の子孫であるか等しいとき、かつそのときに限り真である。

【0114】また述語 `anc`、`desc` を用いて、集合演算 `above`、`below` を定義する。

$$above(A, B) = \{a \in A \mid (\exists b \in B) (anc(a, b))\}$$

$$below(A, B) = \{a \in A \mid (\exists b \in B) (desc(a, b))\}$$

ここで、 A, B は集合である。

`above(A, B)` を求めることは、 A の要素のうち B のどれかの祖先であるか等しいものを選択する操作に等しい。`below(A, B)` を求めることは、 A の要素のうち B のどれかの子孫であるか等しいものを選択す

る操作に等しい。

【0115】更に、述語 `anc`、`desc` を用いて、集合演算 `top`、`bottom` を定義する。

$$\begin{aligned} \text{top}(A) &= \{a \in A \mid (\forall b \in A) (a=b \vee \neg \text{anc}(b, a))\} \\ \text{bottom}(A) &= \{a \in A \mid (\forall b \in A) (a=b \vee \neg \text{desc}(b, a))\} \end{aligned}$$

$\text{top}(A)$ を求めることは、 A の要素のうち他の要素の子孫になっていないものだけを選択する操作に等しい。 $\text{bottom}(A)$ を求めることは、 A の要素のうち他の要素の先祖になっていないものだけを選択する操作に等しい。

$$\begin{aligned} \text{mother}(a, b) &\Leftrightarrow \text{above}(a, b) \wedge \text{length}(a) = \text{length}(b) - 1 \\ \text{daughter}(a, b) &\Leftrightarrow \text{below}(a, b) \wedge \text{length}(a) + 1 = \text{length}(b) \\ \text{parent}(A, B) &= \{a \in A \mid (\exists b \in B) (\text{mother}(a, b))\} \\ \text{child}(A, B) &= \{a \in A \mid (\exists b \in B) (\text{daughter}(a, b))\} \end{aligned}$$

ここで、 A, B は集合である。

$\text{parent}(A, B)$ を求めることは、 A の要素のうち B のどれかの親であるか等しいものを選択する操作に等しい。 $\text{child}(A, B)$ を求めることは、 A の要素のうち B のどれかの子であるか等しいものを選択する操作に等しい。

【0117】図4に示す木構造に、行きがけ順 (preorder) に番号を振ったものを図7に示す。構造化文書の論理構造においては、ほとんどの場合、行きがけ順に従ってレイアウトされるので、この順番は特別な意味を持つ。例えば図4の論理構造は、通常、第1章、1. 1節、1. 2節、1. 3節、第2章、2. 1節、2. 2節、2. 2. 1節、2. 2. 2節、2. 2. 3節、の順にレイアウトされる。

【0118】図4に示す木構造の各ノードのアドレスを行きかけ順に並べると、〈1〉、〈11〉、〈111〉、〈112〉、〈113〉、〈12〉、〈121〉、〈122〉、〈1221〉、〈1222〉、〈123〉となる。

【0119】2つのアドレスのうち行きがけ順でどちら

$$\begin{aligned} \text{head}(A, B) &= \{a \in A \mid (\exists b \in B) (\text{pred}(a, b))\} \\ \text{tail}(A, B) &= \{a \in A \mid (\exists b \in B) (\text{succ}(a, b))\} \end{aligned}$$

ここで、 A, B は集合である。

$\text{head}(A, B)$ を求めることは、 A の要素のうち B のどれかに等しいか、行きがけ順でそれよりも早く出現するものを選択する操作にひとしい。 $\text{tail}(A, B)$ を求めることは、 A の要素のうち B のどれかに等しいか、行きがけ順でそれよりも後に出現するものを選択する操作に等しい。

【0123】上述の様に、接続関係 (木構造における子孫関係、行きがけ順での順序関係) に基づく演算が定義でき、それによって処理を行う識別子オペレータを実現することができる。

【0124】いま、 T を与えられた木構造に属する構成子の識別子の全体として、

$T \setminus \text{below}(T, A)$ は A の各要素を頂点とする部
 $A \subset T$ とすると、

しい。

【0116】同様にして述語 mother 、 daughter を定義し、これらを用いて集合演算 parent 、 child を定義できる。

が先にあるかをアドレスが持つ情報だけから判定することができ。アドレス addr の i 番目の要素を addr_i で表すことにする。 $i > \text{len}(\text{addr})$ のときには、 $\text{addr}_i = 0$ と定める。

【0120】ここで述語 pred を次のように定義する。

$$\text{pred}(a, b) \Leftrightarrow a_{\text{lub}(a, b)+1} \leq b_{\text{lub}(a, b)+1}$$

$\text{pred}(a, b)$ は、 a, b のうち行きがけ順で先に表されるものを求める関数である。

【0121】同様に、述語 succ を次のように定義する。

$$\text{succ}(a, b) \Leftrightarrow a_{\text{lub}(a, b)+1} \geq b_{\text{lub}(a, b)+1}$$

$\text{succ}(a, b)$ は a, b のうち行きがけ順で後に表されるものを求める関数である。

【0122】そして述語 pred 、 succ を用いて、集合演算 head 、 tail を定義する。

分木を、もとの木構造から除去する操作に等しい。このような操作を行う構成子アクセサを実現することができる。

【0125】次に第1の実施例の処理動作について、図8乃至図11を参照して説明する。なお、図8は全体の処理の流れを示すメインルーチンを示し、図9はその処理において実行されるスクリプトの実行処理を示すサブルーチンを示し、図10はそのスクリプトの実行処理において実行される基本単位の処理を示すサブルーチンを示し、図11はその実行単位の処理において実行される基本実行単位の処理を示すサブルーチンを示している。

【0126】ここで、実行単位と基本実行単位を次の様に定義する。

【0127】 a, b, c, \dots, z をコマンドとする。単

独のコマンドは基本実行単位である。コマンドとその引数も基本実行単位である。コマンド（とその引数）の入力、出力の一方あるいは両方をリダイレクト（切り替え）したものも基本実行単位である。基本実行単位は実行単位である。コマンドをセミコロンで繋いだもの、例えば、

a ; b

は個々のコマンドがそれぞれ1つの基本実行単位である。つまり、上記の例では、a、bの2つの基本実行単位がある。コマンドをパイプでつないだもの、例えば、

a | b

は、全体で1つの実行単位である。実行単位を対になった丸括弧でくくったもの、例えば、

(a ; b)

は、1つの基本実行単位である。より複雑な例として、

(a ; b) | c

を考えると、これ全体で1つの実行単位である。

【0128】さて全体の処理の流れについて説明する。図8に示すように、シェル（コマンドインタプリタ）は、スクリプトの実行可否かを判断し（ステップ801）、スクリプトの実行の場合には、そのスクリプトを実行する（ステップ802）。

【0129】一方、ステップ801においてスクリプトの実行でない場合は、スクリプトを実行単位に分割し（ステップ803）、未実行の実行単位が存在するかどうかを判断する（ステップ804）。ここで、未実行の実行単位が存在する場合は、最も左の実行単位を取り出し（ステップ805）、この実行単位を処理し（ステップ806）、その後、未実行の実行単位が存在するかどうかを判断する（ステップ807）。ここで、未実行の実行単位が存在する場合は、上記ステップ805に戻りこのステップ以降を実行する。

【0130】なお、ステップ804、807において、未実行の実行単位が存在しない場合はリターンする。

【0131】次に、スクリプトの実行処理について説明する。

【0132】図9に示すように、シェルは、ファイルの終りであるかを判断し（ステップ901）、ファイルの終りでない場合はバッファをクリアし（ステップ902）、スクリプト中の1行を読み込んで、バッファに追加（格納）すると共に（ステップ903）、その読み込んだ行の最後のトークンが“\”であるかどうかを判断する（ステップ904）。

【0133】ステップ904において、“\”の場合は上記ステップ903に戻りこのステップ以降を実行し、反対に、“\”でない場合は変数を展開すると共に（ステップ905）、スクリプトを実行単位に分割し（ステップ906）、更に未実行の実行単位は存在するかどうかを判断する（ステップ907）。

【0134】ステップ907において、未実行の実行単

位が存在する場合は、最も左の実行単位を取り出し（ステップ908）、該実行単位を処理し（ステップ909）、未実行の実行単位はあるか否かを判断する（ステップ910）。

05 【0135】ステップ910において、未実行の実行単位が存在する場合は上記ステップ908に戻りこのステップ以降を実行し、反対に未実行の実行単位が存在しない場合はファイルの終りであるかを判断する（ステップ911）。

10 【0136】なおステップ907において未実行の実行単位が存在しない場合はステップ911に進む。

【0137】ステップ911において、ファイルの終りでない場合は上記ステップ902に戻りこのステップ以降を実行する。ステップ911、ステップ901においてファイルの終りの場合は処理を終了し、図8のステップ802にリターンする。

【0138】次に、実行単位の処理について説明する。

【0139】図10に示すように、シェルは、実行単位を基本実行単位に分割し（ステップ1001）、パイプ、リダイレクションの処理を行う（ステップ1002）。

【0140】次に未実行の基本実行単位が存在するかどうかを判断し（ステップ1003）、未実行の基本実行単位がある場合は、最も左の基本実行単位を取り出し（ステップ1004）、この基本実行単位の処理を実行し（ステップ1005）、その後、未実行の基本実行単位が存在する否かを判断する（ステップ1006）。

【0141】ステップ1006において、未実行の基本実行単位が存在する場合は、上記ステップ1004に戻りこのステップ以降を実行する。なおステップ1006及びステップ1003において、未実行の基本実行単位が存在しない場合は、処理を終了し、リターンする。ここで、この実行単位の処理が、スクリプトの実行処理のサブルーチンとして実行されたときは図8のステップ802にリターンし、コマンドの実行処理のサブルーチンとして実行されたときは図9のステップ909にリターンする。

【0142】次に、基本実行単位の処理について説明する。

40 【0143】図11に示すように、シェルは、該当する基本実行単位における最も左のトークンが“（”であるかどうかを判断し（ステップ1101）、“（”である場合は基本実行単位から左右の丸括弧を除去した文字列を引数として、サブシェルを起動し（ステップ1102）、一方、“（”でない場合はコマンドを起動する（ステップ1103）。

45 【0144】ステップ1102において、サブシェルが起動されると、このサブシェルによって、図8～図11に示される処理手順が実行される（つまり図8～図11に示される処理が再帰的に実行される）。

50 【0144】ステップ1102、1103が終了した場

合は、図10のステップ1005にリターンする。

【0145】なお、スクリプトの実行には既存のシェルの用いることができる。

【0146】ここで、図12、図13に示される様な、処理対象となる文書を用いて文書処理を説明する。

【0147】なお図12は図の論理構造を示したものであり、図13は章の論理構造を示したものであり、これら各図において、矩形内に示されている文字列が属性“uobject-type”の値として構成子に指定されているものとする。

【0148】また図12に示した論理構造では、図は、グラフィックスとオプションな図見出しからなっている。“Figure”と記述された構成子が、図を表す論理構造の根である。また“Raster Graphics”と記述された構成子がグラフィックスを含む内容部である。また“Caption Number”と記述された構成子が、図見出し番号を持つ内容部である。更に“Caption Number”と記述された構成子が、図見出しのテキストを持つ内容部である。

【0149】更に図13に示した論理構造では、章（あるいは節）は、見出し、段落の繰り返し、オプションな部分節からなっている。従って、章（あるいは節）は入れ子になって良い。“Numbered Segment”と記述された構成子が、章（節）を表す論理構造の根である。“Title”と記述された構成子が見出しを表す部分木の根であ

```
#!/bin/sh
```

```
(select-type -e Figure $1; select-type -e Title $1) | union | \
extract -s - $1
```

このスクリプトは指定された文書の図と見出しを検索する例である。引数は検索対象となる文書名である。複数の引数が与えられた場合、2つ目以降の引数は無視される。

【0154】最初のselect-type コマンドの実行により、指定されたファイル中の構成子記述列のうち、属性“uobject-type”の値が“Figure”を含む構成子（すなわち図を表す部分木の根）の識別子列が求められる。同様に、2番目のselect-type コマンドの実行により、属性“uobject-type”の値が“Title”を含む構成子（すなわち見出しの部分木の根）の識別子列が求められる。union コマンドの実行により、最初のselect-type コマンドの実行と2番目のselect-type コマンドの実行とによって得られた2つの識別子列（集合）の和集合が得られる。最後にextract コマンドの実行により、union コマンドの実行により得られた識別子列に含まれる識別子を持つ構成子を頂点とする部分木の構成子記述列がファイルから抜き出される。

【0155】上述したように第1の実施例によれば、指定された構造化文書から条件に適合する文書部品を検索することができる。

【0156】次に本発明の第2の実施例を図14を参照

る。“Title Number”と記述された構成子が、章（節）見出し番号をもつ内容部である。“Title Text”と記述された構成子が、章（節）見出しのテキストを持つ内容部である。

05 【0150】次に、実際に処理を行うためのコマンドの仕様例、及びそのコマンドを用いたスクリプト（コマンドプロシジャ）の例を以下に示す。すなわちBourneシェルのスクリプトである（Bourneシェルの仕様についてはUNIXのオンラインマニュアルを参照）。

10 【0151】セミコロンで区切られたコマンドは順次実行される。丸括弧（すなわち“(”や“)”）で囲まれたコマンドの列はサブシェルで実行され、あたかも単独のコマンドであるかの様に扱われる。行末のバックスラッシュは、次の行が継続行であることを示す。

15 【0152】\$1、\$2などは、スクリプトに与えられた引数を参照するための変数（positional parameter）である（具体的には文書名つまりファイル名である）。例えばスクリプト名がfooであるとして、コマンドラインから

20 % foo who are you
と入力したとすると、\$1はwho、\$2はare、\$3はyouである。

【0153】◇ [スクリプト例1]

して説明する。

30 【0157】図14は本発明に係る文書処理装置の第2の実施例を機能ブロック図で示したものである。この機能ブロック図は、図1に示した第1の実施例の機能ブロック図の構成において、構成子記述列取得部150を削除し、構成子記述列挿入部1200を追加した構成になっている。なお同図において、図1に示した構成要素と同様の機能を果たす部分には同一の符号を付している。

35 【0158】構成子記述列挿入部1200では、部品挿入部1210が、第2の識別子列取得部140により得られた識別子列に対応する構成子記述列を、指定された条件としての条件情報1220に従って、構成子記述列保持部110に保持されている構成子記述列群に挿入する。なお条件情報1220は文書部品を挿入するための条件を示す情報である。部品挿入部1210による挿入処理後の構成子記述列は、既に保持されている他の構成子記述列と区別できるように構成子記述列保持部110に保持され、さらに処理操作の対象にもなり得るものである。

40 【0159】この第2の実施例においても、構成子記述列挿入部1200は、この構成要素の機能を遂行させるためのソフトウェア（プログラム）を制御手段が実行す

ることにより実現される。従って、部品挿入部 1 2 1 0 は上述した部品挿入のコマンド（つまり insert コマンド）に相当する。

【0160】なおこの第 2 の実施例では、第 2 の識別子列取得部 1 4 0 の識別子オペレータ 1 4 1 は上述した bottom コマンドに相当するように設定されている。またスクリプトを実行するシェルの処理は、上記第 1 の実施例の図 8 乃至図 1 1 で説明した処理と同様である。

【0161】次に、実際に処理を行うためのコマンドの仕様例、及びそのコマンドを用いたスクリプトの例を以下に示す。

【0162】◇ [スクリプト例 2]

```
#!/bin/sh
```

```
select-type -e Segment $2 | bottom > pid
insert -yc pid $1 $2
```

このスクリプトは、部品として挿入されるファイル（以下、部品ファイルという）中の構成子記述列を他のファイル（以下、文書ファイルという）中の構成子記述列に含まれる最下位の節のうち先頭にあるものの末子になるように挿入される例である。

【0163】このスクリプトの第 1 引数は部品として挿入される部品ファイル、第 2 引数は部品が挿入される文書ファイルである。3 個以上の引数が与えられた場合、3 つ目以降の引数は無視される。

【0164】select-type コマンドの実行により、文書ファイルの構成子のうち、属性 “uobject-type” の値が “Segment” を含む構成子（すなわち節を表す部分木の根）の識別子列が求められる。次に、bottom コマンドの実行により、select-type コマンドの実行により得られた識別子列のうち最下位にある構成子の識別子列だけが得られる。bottom コマンドの実行による出力は、作業用のファイル pid に格納される。最後に、insert コマンドの実行により、部品ファイルの構成子記述列が、文書ファイルに挿入される。挿入位置は、ファイル pid 中の先頭の識別子に対応する構成子の末子である。

【0165】上述した様に第 2 の実施例によれば、ある構造化文書中の文書部品を抽出し、この文書部品を他の

```
#!/bin/sh
```

```
(select-type -e Figure $1 ; select-type -e Segment $1) | union | \
chval -n Patent 特許 - $1 > $1.$$
mv $1.$$ $1
```

このスクリプトは、指定された文書の図或いは節を表す部分木の根の属性 “user visible name” に現れる文字列 “Patent” を “特許” に変更する例である。

【0173】変数 \$\$ は、このスクリプトが実行されたときの /bin/sh のプロセス番号に展開される。

【0174】2 つの select-type コマンドは、第 1 の実施例のスクリプト例 1 における select-type コマンドと同様の目的で用いられる。union コマンドの実行によ

構造化文書中に挿入することができる。

【0166】次に本発明の第 3 の実施例を図 1 5 を参照して説明する。

【0167】図 1 5 は本発明に係る文書処理装置の第 3 の実施例を機能ブロック図で示したものである。この機能ブロック図は、図 1 に示した第 1 の実施例の機能ブロック図の構成において、構成子記述列取得部 1 5 0 を削除し、構成子記述列変更部 1 3 0 0 を追加した構成になっている。なお同図において、図 1 に示した構成要素と同様の機能を果たす部分には同一の符号を付している。

【0168】構成子記述列変更部 1 3 0 0 では、部品変更部 1 3 1 0 が、第 2 の識別子列取得部 1 4 0 により得られた識別子列に対応する構成子記述列を、文書部品の属性を変更するための条件を示す情報である指定された変更指示情報 1 3 2 0 に従って変更する。この結果である変更処理後の構成子記述列は、既に保持されている他の構成子記述列と区別できるように構成子記述列保持部 1 1 0 に保持され、さらに処理操作の対象にもなり得るものである。

【0169】この第 3 の実施例においても、構成子記述列変更部 1 3 0 0 は、この構成要素の機能を遂行させるためのソフトウェア（プログラム）を制御手段が実行することにより実現される。従って、部品変更部 1 3 1 0 は上述した部品変更のコマンド（つまり chval コマンド）に相当する。

【0170】なおこの第 3 の実施例では、第 1 の識別子列取得部 1 3 0 の構成子レゾルバ 1 4 1 は上述した select-type コマンドに相当し、第 2 の識別子列取得部 1 4 0 の識別子オペレータ 1 4 1 は上述した union コマンドに相当するように設定されている。またスクリプトを実行するシェルの処理は、上記第 1 の実施例の図 8 乃至図 1 1 で説明した処理と同様である。

【0171】次に、実際に処理を行うためのコマンドの仕様例、及びそのコマンドを用いたスクリプトの例を以下に示す。

【0172】◇ [スクリプト例 3]

り、2 つの select-type コマンドそれぞれの実行結果である識別子列の和集合が求められる。chval コマンドの実行により、union コマンドの実行によって得られた識別子列で指定された構成子記述列の属性 “user visible name” に含まれる文字列 “Patent” が “特許” に置換される。chval コマンドの実行結果である構成子記述列は、作業用ファイルに格納される。最後に、UNIX の標準コマンド mv の実行によって、作業用ファイルの名前

をもとの文書の名前に付け換えられる。

【0175】上述したように第3の実施例によれば、指定された構造化文書中の文書部品の属性を指定された属性値に変更することができる。

【0176】次に本発明の第4の実施例を図16を参照して説明する。

【0177】図16は本発明に係る文書処理装置の第4の実施例を機能ブロック図で示したものである。この機能ブロック図は、図1に示した第1の実施例の機能ブロック図の構成において、構成子記述列取得部150を削除し、文書名出力部1400を追加した構成になっている。なお同図において、図1に示した構成要素と同様の機能を果たす部分には同一の符号を付している。

【0178】文書名出力部1400では、文書オペレータ1410が、第2の識別子列取得部140により得られた識別子列の識別子数を検査し、この結果が指定された第3の条件としての条件情報1420に適合する場合は、予め記述された文書の文書名を出力する。なお条件情報1420文書名を取得するための条件を示す情報で

```
#!/bin/sh
```

```
(select-type -e Figure $1 ; select-type -e Segment $1) | union | \
id-doc -gt 10 $1
```

このスクリプトは、与えられた文書にある図と節の数を調べ、図と節が計10個以上あったときには、与えられた文書の名前を出力するものである。条件を満たさないときには、何も出力しない。

【0183】select-type コマンドとunion コマンドは、第3の実施例のスクリプト例3におけるselect-type コマンド及びunion コマンドと同様の目的で用いられる。id-docコマンドの実行により、union コマンドの実行によって得られた識別子の数を検査し、この検査結果が10よりも多い場合は、与えられた文書名が標準出力に書き出される。

【0184】上述したように第4の実施例によれば、指定された条件に適合する構造化文書の文書名のみを得ることができる。

【0185】次に本発明の第5の実施例を図17を参照して説明する。

【0186】図17は、本発明に係る文書処理装置の第

```
#!/bin/sh
```

```
select-type -e Segment $2 | bottom > pid
select-type -e Figure $1 | extract -s - $1 | \
insert -yc pid - $2
```

このスクリプトは、二つの文書を入力として、第1引数で指定された文書に含まれる図を、第2引数で指定された文書の最下位の節のうち最初のものの末子として挿入するものである。

【0190】実行文の1行目は、第3の実施例のスクリプト例3で説明した通りである。2番目のselect-type コマンドは、第3の実施例のスクリプト例3における最

ある。

【0179】この第4の実施例においても、文書名出力部1400は、この構成要素の機能を遂行させるためのソフトウェア（プログラム）を制御手段が実行することにより実現される。従って、文書オペレータ1410は上述した文書オペレータのコマンド（つまりid-doc）に相当する。

【0180】なおこの第4の実施例では、第1の識別子列取得部130の構成子レゾルバ141は上述したselect-type コマンドに相当し、第2の識別子列取得部140の識別子オペレータ141は上述したunion コマンドに相当するように設定されている。またスクリプトを実行するシェルの処理は、上記第1の実施例の図8乃至図11で説明した処理と同様である。

【0181】次に、実際に処理を行うためのコマンドの仕様例、及びそのコマンドを用いたスクリプトの例を以下に示す。

【0182】◇[スクリプト例4]

5の実施例を機能ブロック図で示したものである。この機能ブロック図は、第1～第4の実施例の機能ブロック図を合成した構成になっている。すなわち、図1に示した第1の実施例の構成に、図14に示した第2の実施例の機能ブロック図の構成子記述列挿入部1200、図15に示した第3の実施例の機能ブロック図の構成子記述列変更部1300、図16に示した第4の実施例の機能ブロック図の文書名出力部1400を追加した構成になっている。

【0187】なおこの第5の実施例では、スクリプトを実行するシェルの処理は、上記第1の実施例の図8乃至図11で説明した処理と同様である。

【0188】次に、実際に処理を行うためのコマンドの仕様例、及びそのコマンドを用いたスクリプトの例を以下に示す。

【0189】◇[スクリプト例5]

初のselect-type コマンドと同様の目的で使用される。extract コマンドの実行により、第1引数で指定された文書の図を表す構成子記述列が抜き出される。最後に、extract コマンドの実行によって得られた構成子記述列を、insert コマンドの実行によって、第2引数で指定された文書に挿入する。挿入は、第2の実施例のスクリプト例2と全く同様に行われる。

【0191】◇ [スクリプト例6]

```
#!/bin/sh
select-type -e Segment $2 | bottom > pid
select-type -e Figure $1 | chval -n Patent 特許 - $1 | \
insert -yc pid - $2
```

このスクリプトは、スクリプト例5とほぼ同じ働きをするが、図を表す部分木の根の構成子の属性値を更新するものである。

【0192】実行文の1行目は第3の実施例のスクリプト例3で説明した通りである。2番目のselect-type コマンドは第2の実施例のスクリプト例2における最初のselect-type コマンドと同様の目的で使用される。chva

```
#!/bin/sh
(select-type -e Figure $1 ; select-type -e Segment $1) | union | \
extract -s $1 > part
echo -n > empty
chval -n Patent 特許 empty part
```

このスクリプトは、図あるいは節を表す構成子記述列を検索した後、構成子の属性“user visible name”の値を変更するものである。

【0194】select-type コマンド、union コマンド、extract コマンドにより、引数で指定された文書から図あるいは節を表す部分木が検索され、対応する構成子記述列が作業用ファイルpartに格納される。次に、UNI

```
#!/bin/sh
select-type -e Segment $2 | bottom > pid
select-type -e Figure $1 > fig
extract -s fig $1 | chval -n Patent 特許 fig $1 | \
insert -yc pid - $2
```

このスクリプトは、上記スクリプト例7の例とほぼ同じ働きをする。なお、このスクリプト例8においては、最終的に挿入を行う位置を、上記スクリプト例6の例と同

```
#!/bin/sh
select-type -e Figure $1 | extract -s - $1 | \
sed -e "user-visible-name/s/Patent/特許/g"
```

この例では、テキスト操作を行うUNIXの標準コマンドsedを用いて、検索結果得られた構成子記述列中の属性“user visible name”の値に含まれる文字列“Patent”を“特許”に置換するものである。

【0197】次に、図12に示した図の論理構造を有す

```
% select-type -e Figure test.doc | extract -s - test.doc | \
insert -yc -p "3" - skeleton.doc > result.doc
```

最初に、select-type コマンドの実行により、test.docなるファイルから、属性“uobject-type”の値に部分文字列“Figure”を含む構成子の識別子列を列挙し、標準出力に書き出される。次に、extract コマンドの実行により、標準入力から識別子列が読み込まれ、各識別子を根とする部分木に対応する構成子記述列がファイルtest.docから抜き出される。最後にinsertコマンドの実行により、抜き出された部分木が標準入力から読み込ま

1 コマンドは、第3の実施例のスクリプト例3におけるchval コマンドと同様に用いられる。最後に、insertコマンドの実行により、上記スクリプト例5におけるinsertコマンドの実行による挿入と同様の処理が行われる。

【0193】◇ [スクリプト例7]

Xの標準コマンドecho を引数なしで使用することにより、空きのファイルempty が生成される。最後にchval コマンドの実行により、ファイルpartに格納された全構成子記述列を操作対象として、属性“user visible name”の値に出現する文字列“Patent”が“特許”で置換される。

【0195】◇ [スクリプト例8]

30 様に決定している点が、上記スクリプト例7と異なる。

【0196】◇ [スクリプト例9=テキスト処理機能との組み合わせの例]

る文書に対して処理を行う場合について説明する。

【0198】ある文書から図だけを検索して、それらだけから文書を生成するためのスクリプトの例を以下に示す。

40 【0199】◇ [スクリプト例10]

れ、この部分木がファイルskeleton.docに繋ぎ込まれ、こうして得られた構成子記述列が標準出力に書き出される。ここで、ファイルskeleton.docは文書のヘッダ部分と特定論理根だけからなる。結果として得られた構成子記述列は、result.docなるファイルにリダイレクトされ、出力文書となる。

【0200】次に、コマンドプロシジャ例9の修正版で、見出しに“viewpoint”という文字列を含む図を検索

して、そのような図だけからなる文書を生成するための
スクリプト例を以下に示す。

```
% (select-type Figure test.doc ;select-content\
viewpoint test.doc) |above |extract -s - test.doc |\
insert -yc -p "3" - skeleton.doc > result.doc
```

ファイルskeletonは上記スクリプト例10と同様である。この例では、select-contentコマンドは、"viewpoint" という文字列を含む内容部の識別子を列挙するために用いられる。above コマンドの実行により、2つの識別子列が読み込まれ、述語above が成立する構成子の識別子列が出力される。各識別子列は構成子レゾルバ (select-type、select-content) によって生成されたものであり、従ってある述語が成立する構成子の識別子列で

```
#!/bin/sh
(select-type -e Segment test.doc ;\
(select-content -p introduction test.doc ;\
select-type -e title test.doc) |\
below) |\
above |\
bottom|extract -s - test.doc |\
insert -yc -p "3" - skeleton.doc > result.doc
```

below コマンドは2つの構成子記述列を読み込み、述語below が成立する構成子の識別子列を出力する。従って、この例では、属性 "uobject-type" の値が文字列 "title" を含む構成子に從属する内容部のうち、introductionという文字列を含むものを列挙する。

【0204】bottomコマンドは、一つの識別子列を読み込み、述語bottomが成立する構成子記述列を出力する。

【0205】上記各実施例では、文書名を得る方法として、(a) 文書名をスクリプトに記述する。(b) スクリプトに実行する引数として文書名を与える。の2通りの方法があるが、これを、構成子記述列から得るようにしても良い。

【0206】すなわち、図17中点線で示されるように、文書名列取得部1510と、文書名列保持部1520を設ける。

【0207】文書名列取得部1510においては、文書レゾルバ1511が、文書名列を取得するための条件を示す情報である指定された条件情報1512に基づいて、構成子記述列保存部110から文書名(単数又は複数)を取得し、この文書名を文書名列保持部1520に格納する。

【0208】そして文書名出力部1400が、識別子列保持部120に保持されている第2の識別子列取得部140により得られた識別子列の識別子数を検査し、この結果が指定された第3の条件に適合する場合は、文書名列保持部1520に保持されている文書名を出力する。

【0209】上述したように第5の実施例によれば、指定された構造化文書中から指定された条件に適合する文書部品を検索し、この文書部品を他の構造化文書中に挿

【0201】◇ [スクリプト例11]

ある。従ってabove コマンドの実行結果(出力)は見出しに"viewpoint" という文字列を含む図の識別子列である。

【0202】次に、見出しに"introduction" という文字列を含む章(或いは節)に含まれる最下位の節を選び、それらを一つの文書にまとめるためのスクリプト例を以下に示す。

【0203】◇ [スクリプト例12]

入することができる。

【0210】また、指定された構造化文書中から指定された条件に適合する文書部品を検索し、この文書部品の属性を指定された属性値に変更することができる。

【0211】さらに、指定された構造化文書中から指定された条件に適合する文書部品を検索し、この文書部品の属性を指定された属性値に変更し、更に、この文書部品を指定された構造化文書中に挿入することができる。

【0212】上述した各実施例によれば、構成子記述列、識別子列、文書名列のいずれかを扱える文書処理機能(装置)であれば、上述した様な各文書処理を実行することができる。

【0213】

【発明の効果】以上説明したように本発明によれば、予め記述された単数又は複数の文書に対応し、且つ、識別子が付与されている複数の構成子を有する構成子記述列から、指定された第1の条件に適合する構成子に対応する識別子列を取得し、この識別子列から、指定された第2の条件に適合する識別子列を取得し、この第2の条件に適合する識別子列に基づいて、前記予め記述された単数又は複数の文書に対応する構成子記述列に対する所定の処理を実行するようにし、この所定の処理をするに際し、構成子記述列の取得(検索)、挿入、構成子記述列の属性値の変更、文書名の出力、の各処理のうち、いずれかの処理又は複数の処理を実行するようにしたので、指定された文書から条件に適合する文書部品を検索することができるという利点がある。

【0214】また、ある文書中の文書部品を抽出し、この文書部品を、他の文書中に挿入することができるとい

う利点がある。

【0215】また、指定された条件に適合する文書の文書名のみを得ることができるという利点がある。

【0216】また、指定された文書中から指定された条件に適合する文書部品を検索し、この文書部品を、他の文書中に挿入することができるという利点がある。

【0217】また、指定された文書中から指定された条件に適合する文書部品を検索し、この文書部品の属性を、指定された属性値に変更することができるという利点がある。

【0218】また、指定された構造化文書中から指定された条件に適合する文書部品を検索し、この文書部品の属性を指定された属性値に変更し、更に、この文書部品を指定された構造化文書中に挿入することができるという利点がある。

【0219】さらに、構成子記述列、識別子列に基づいて文書処理を行うようにしているので、これらの情報を扱える他の文書処理機能と組み合わせ、文書あるいは文書部品の検索、挿入、合成、更新を正確に処理することができるという効果を奏する。

【図面の簡単な説明】

【図1】本発明に係る文書処理装置の第1の実施例を示す機能ブロック図。

【図2】図1に示した実施例の装置を実現するためのハードウェア構成を示すブロック図。

【図3】ネットワーク構造の一例を示す図。

【図4】構造化文書における論理構造の一例を示す図。

【図5】図4に示す木構造の各ノードにDeweyの10進分類法に従ってアドレスを割り当てた様子を示す。

【図6】共通の祖先のうち、最も若いものを求めるための処理手順を示すフローチャート。

【図7】図4に示す木構造の各ノードに行きがけ順に辿ったときに出現する順番に従って番号を割り当てた様子を示す図。

【図8】全体の処理の流れを示すメインルーチン。

【図9】全体の処理において実行されるスクリプトの実行処理を示すサブルーチン。

【図10】スクリプトの実行処理において実行される基本単位の処理を示すサブルーチン。

【図11】実行単位の処理において実行される基本実行単位の処理を示すサブルーチン。

【図12】構造化文書における図の論理構造の一例を示す図。

【図13】構造化文書における章・節の論理構造の一例を示す図。

【図14】本発明に係る文書処理装置の第2の実施例を示す機能ブロック図。

【図15】本発明に係る文書処理装置の第3の実施例を示す機能ブロック図。

【図16】本発明に係る文書処理装置の第4の実施例を示す機能ブロック図。

【図17】本発明に係る文書処理装置の第5の実施例を示す機能ブロック図。

【図18】従来の文書処理を説明するために使用した構造化文書の論理構造を示す図。

【図19】図18に示す構造化文書に対する処理結果としての出力文書を示す図。

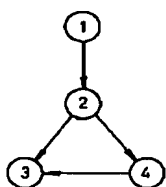
【図20】従来の文書処理を説明するために使用した構造化文書の論理構造を示す図。

【図21】図20に示す構造化文書に対する処理結果としての出力文書を示す図。

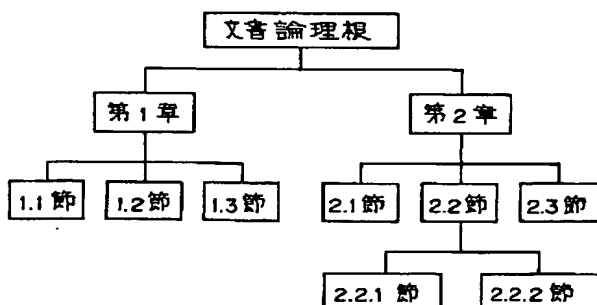
【符号の説明】

110…構成子記述列保持部、120…識別子列保持部、130…第1の識別子列取得部、140…第2の識別子列取得部、150…構成子記述列取得部、210…入力装置、220…ディスプレイ、230…ディスク、240…主メモリ、250…中央処理装置、1200…構成子記述列挿入部、1300…構成子記述列変更部、1400…文書名出力部、1510…文書名列取得部、1520…文書名列取得部。

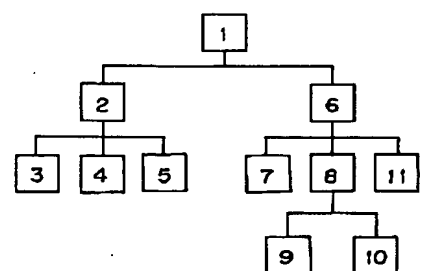
【図3】



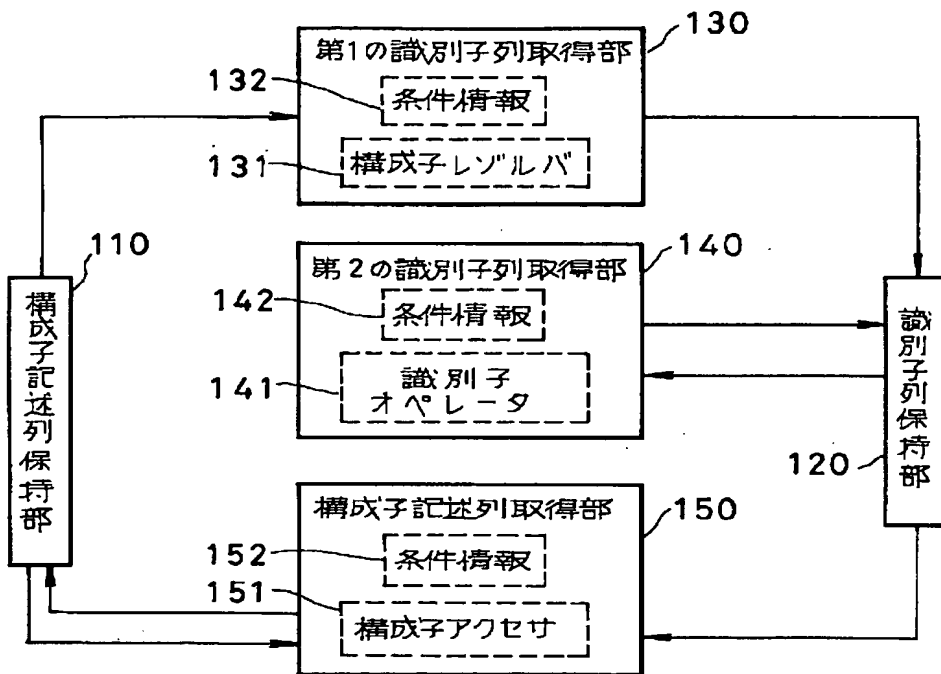
【図4】



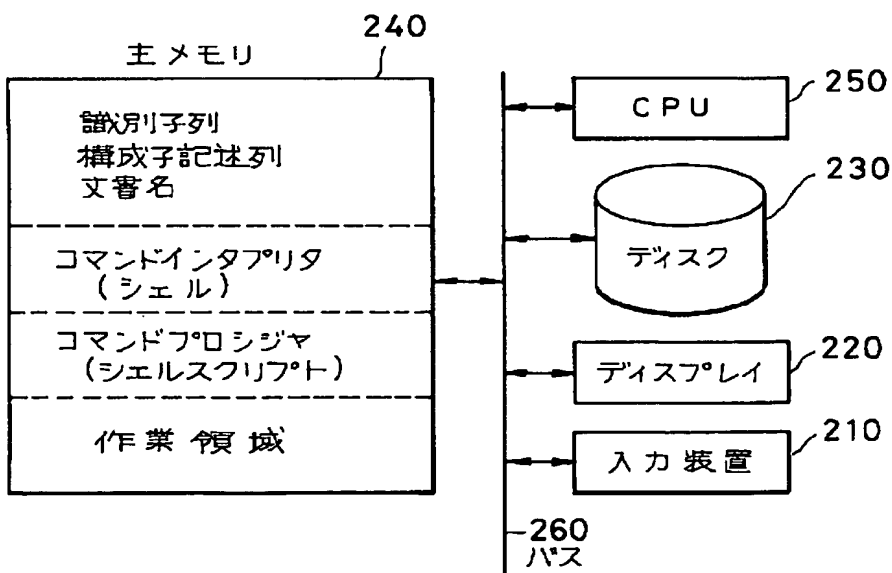
【図7】



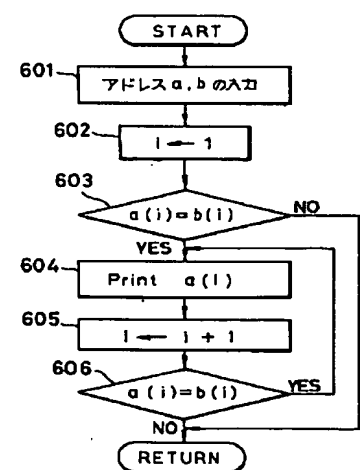
【図 1】



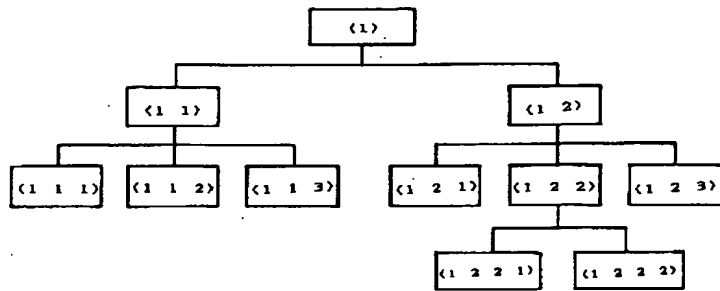
【図 2】



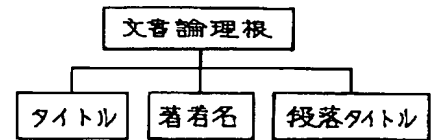
【図 6】



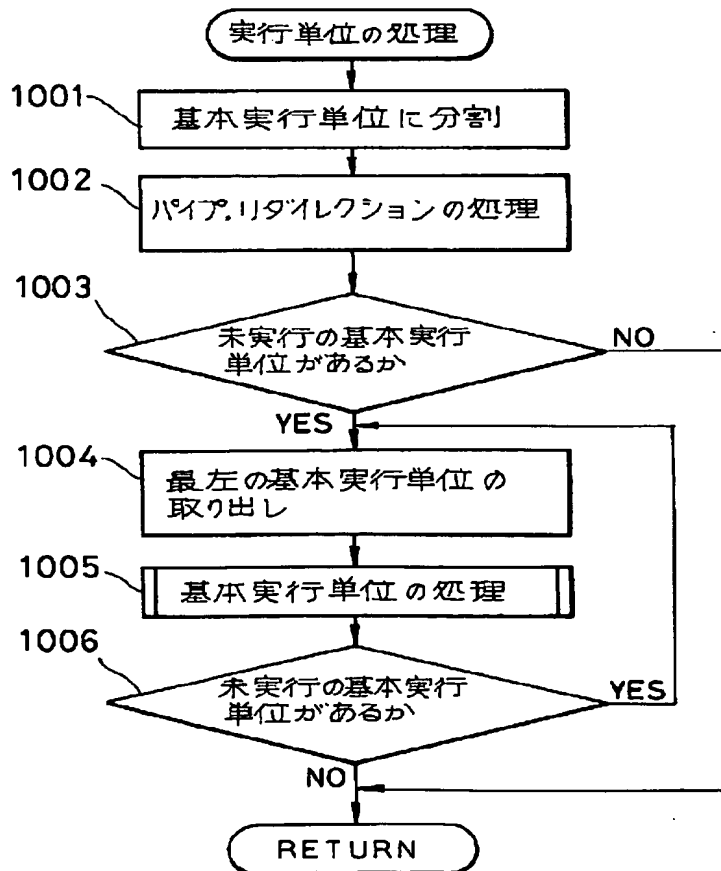
【図 5】



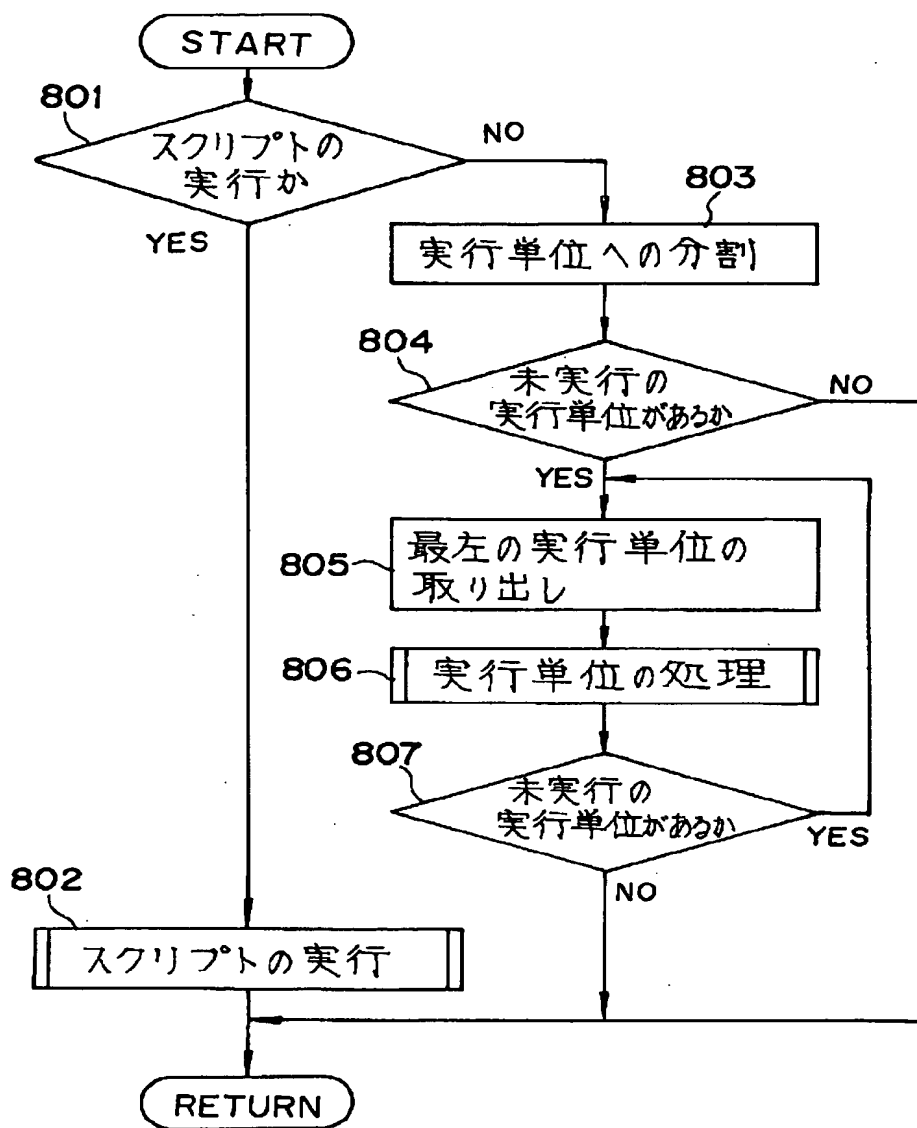
【図 2 1】



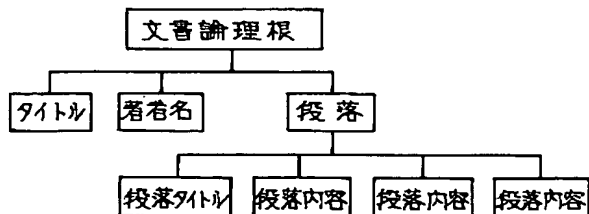
【図 1 0】



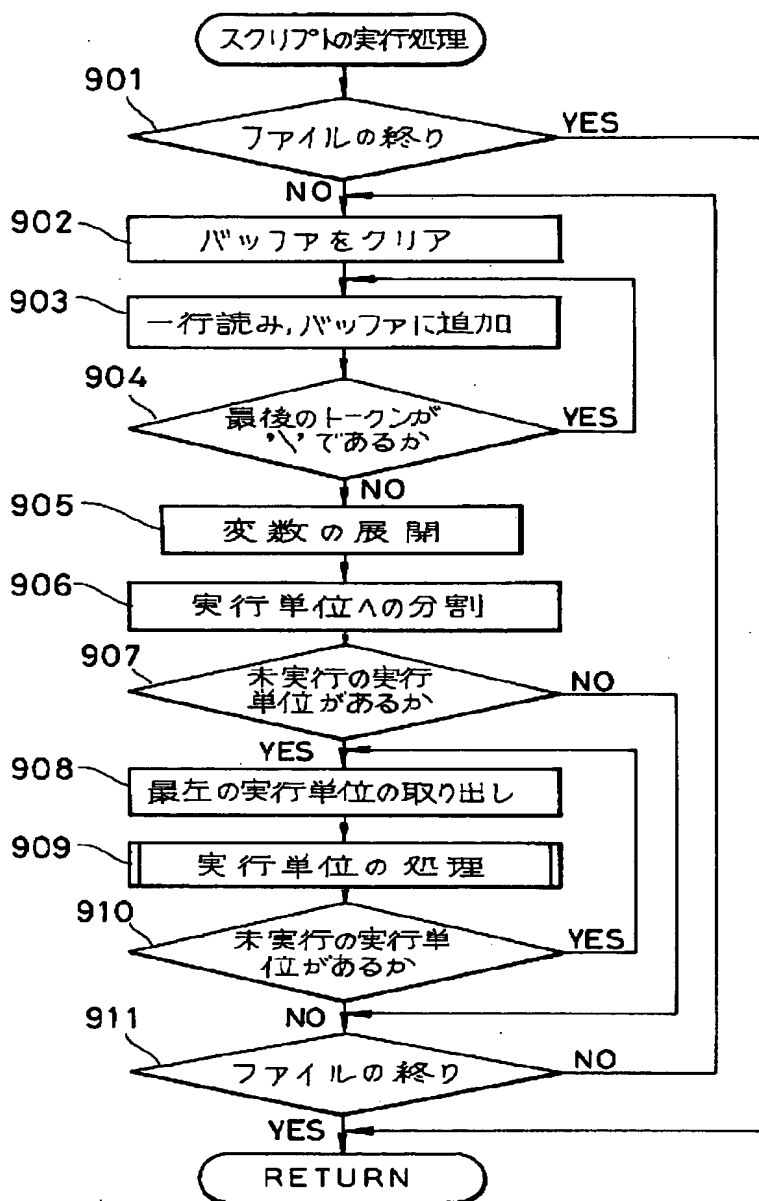
【図 8】



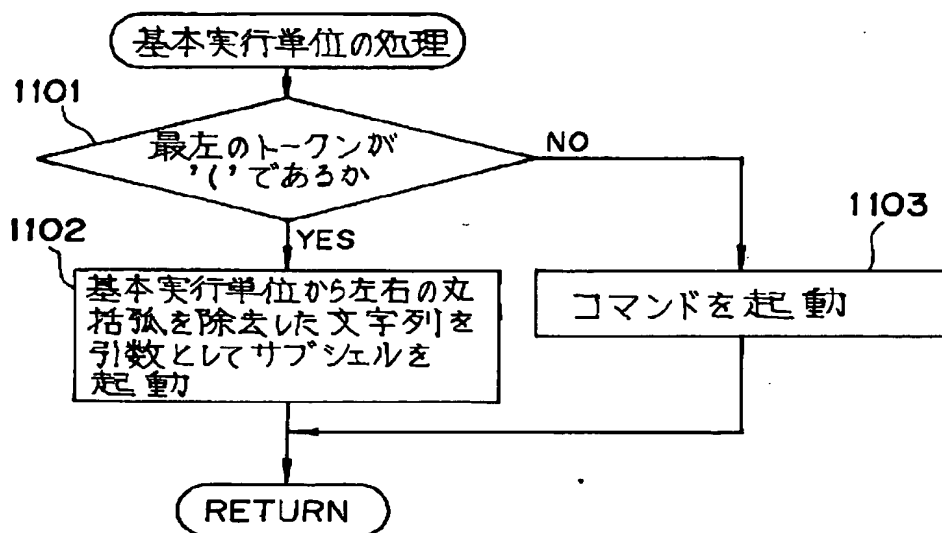
【図 20】



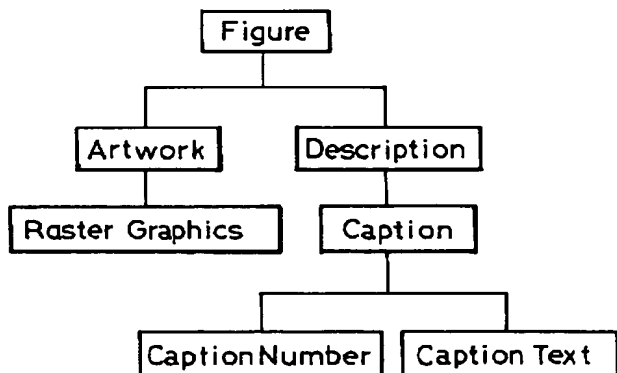
【図9】



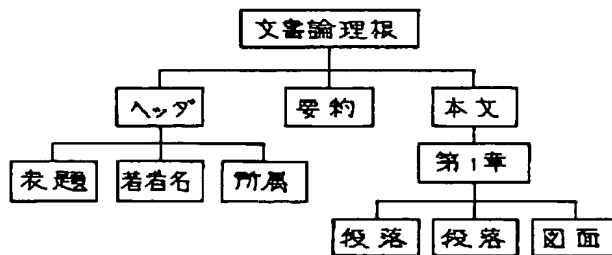
【図 11】



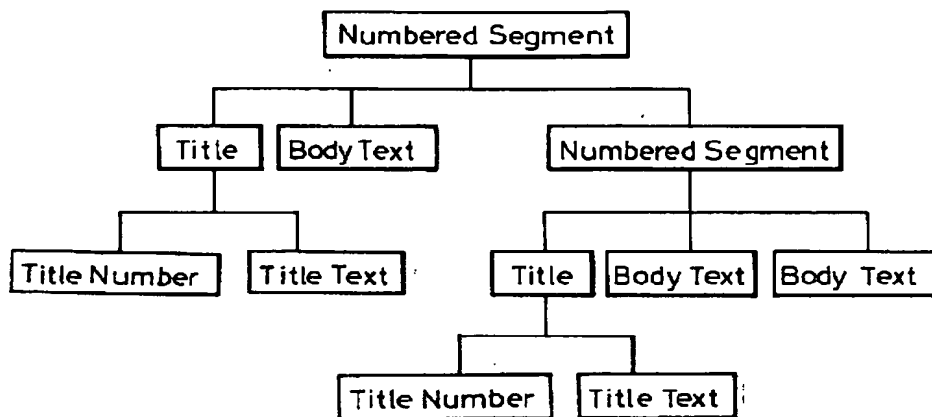
【図 12】



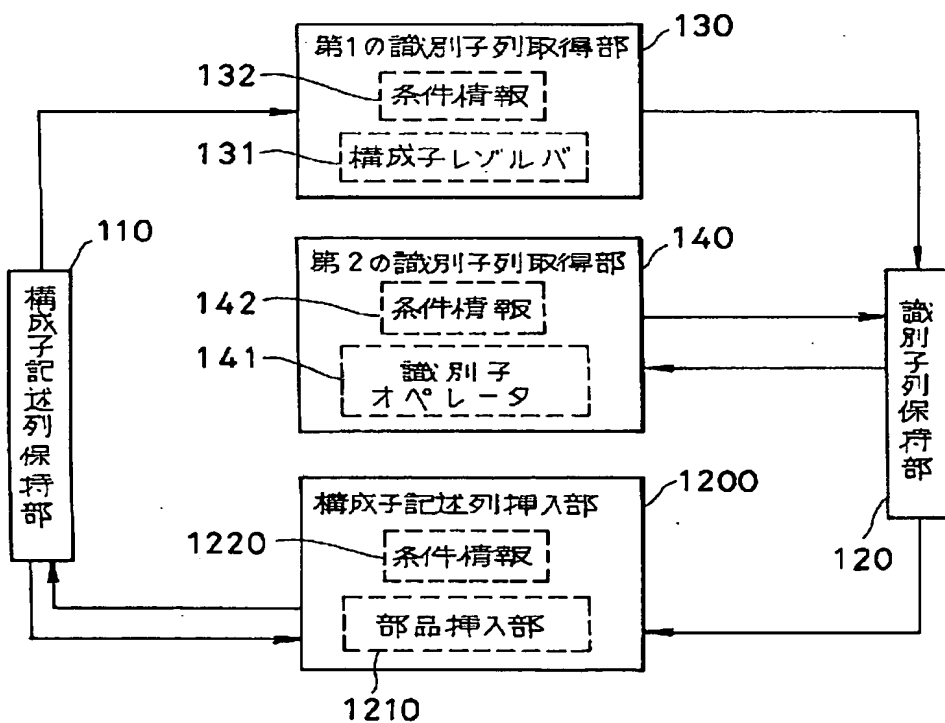
【図 18】



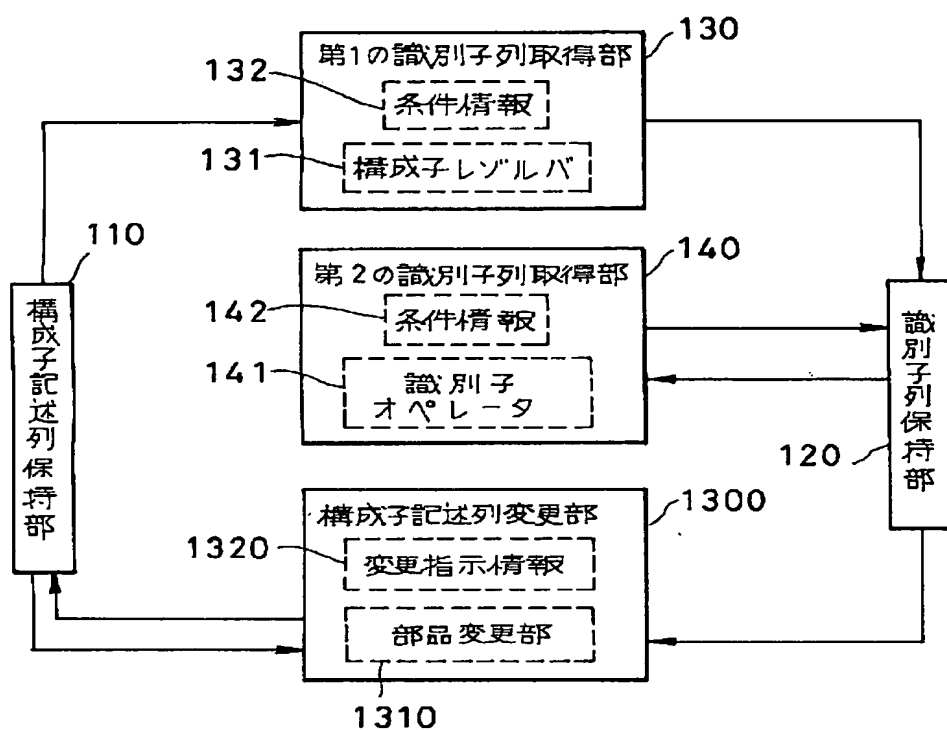
【図 1 3】



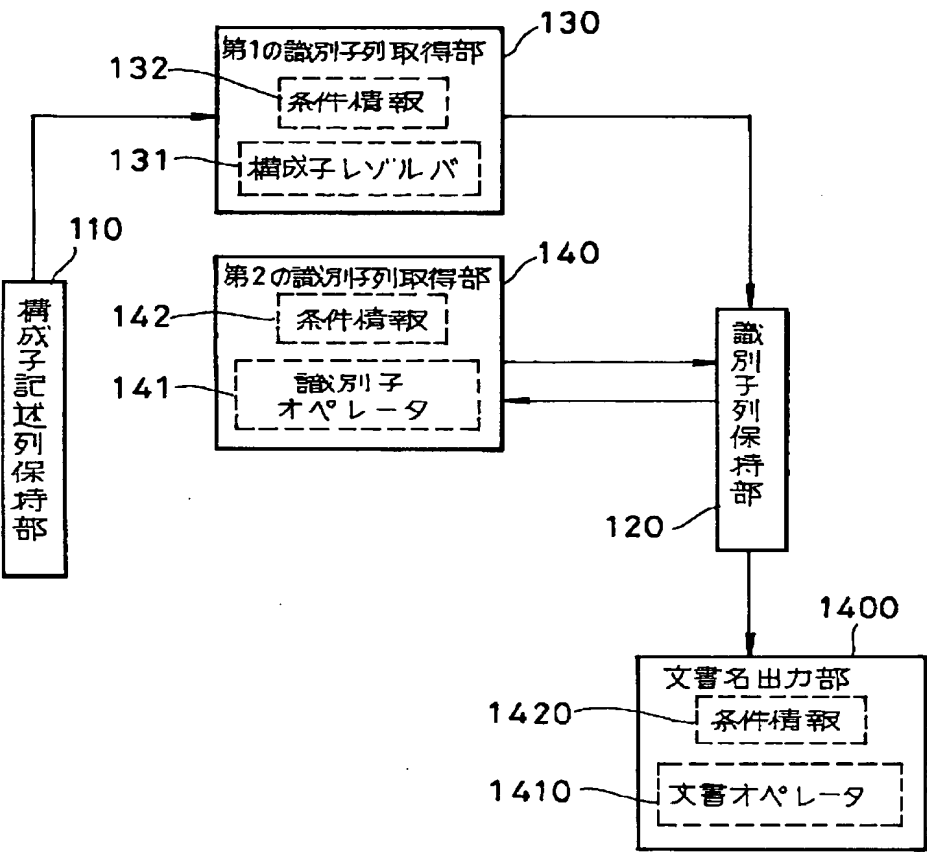
【図 1 4】



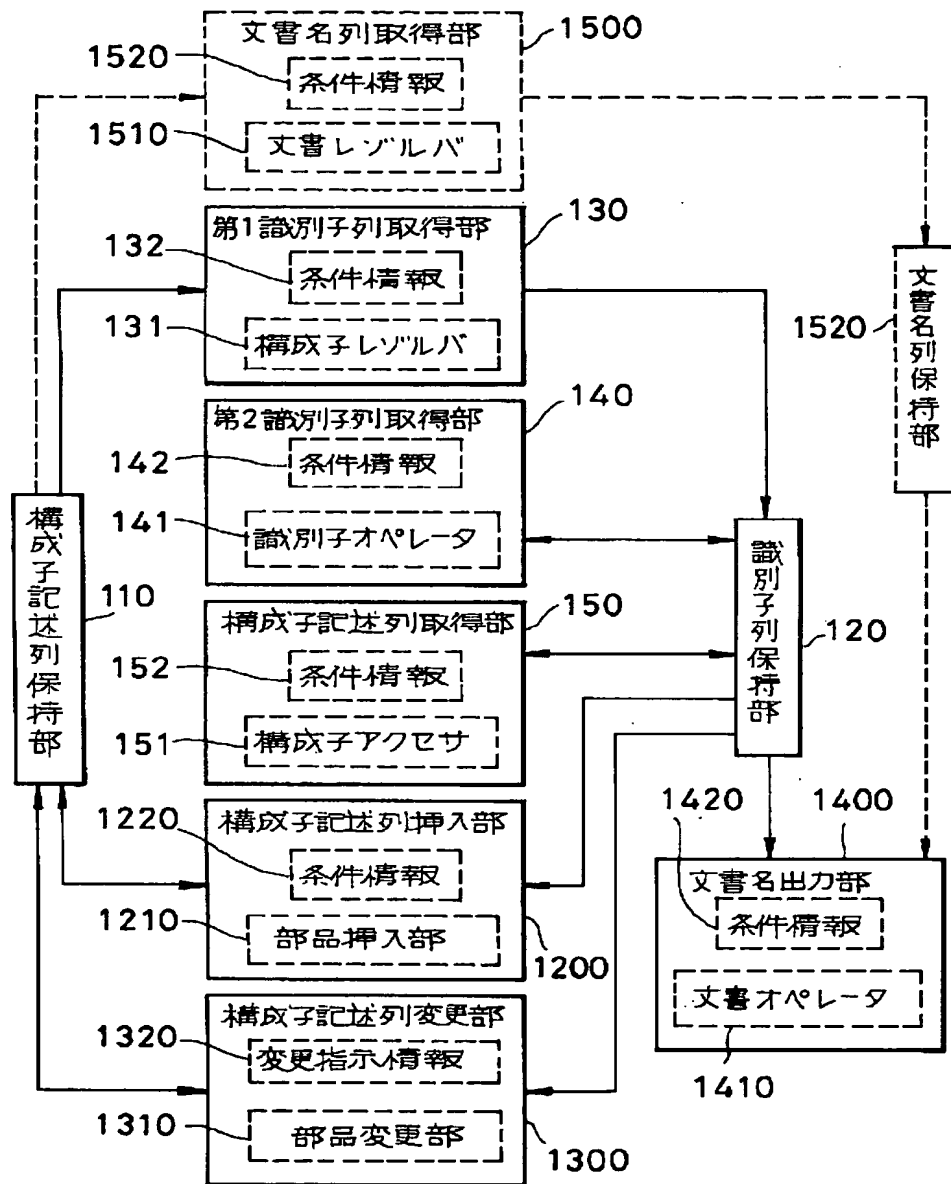
【図15】



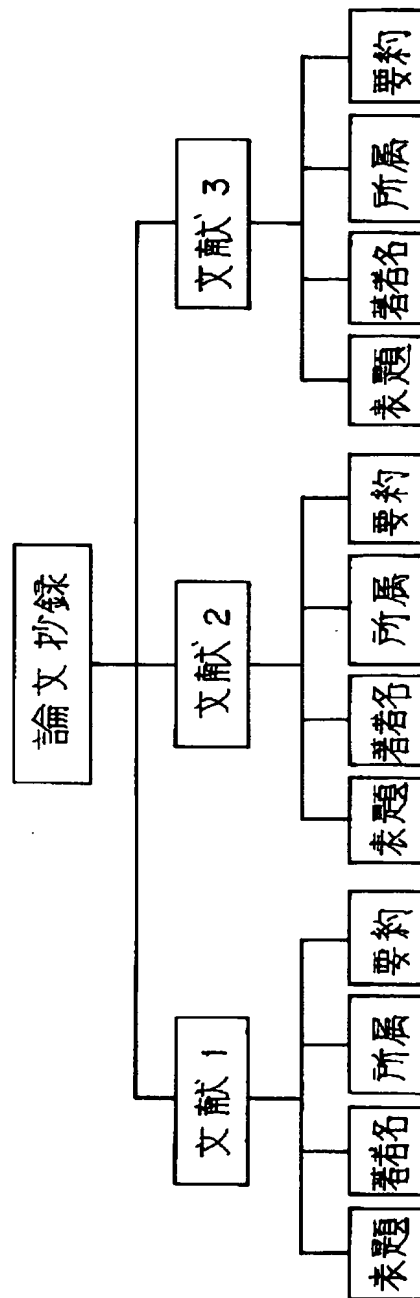
【図 1 6】



【図 17】



【図 19】



【手続補正書】

【提出日】平成 5 年 3 月 1 9 日

【補正方法】変更

【手続補正 1】

【補正内容】

【補正対象書類名】明細書

【0 1 2 0】ここで述語 pred を次のように定義す

【補正対象項目名】0 1 2 0

05 る。

$$\text{pred}(a, b) \Leftrightarrow a_{\text{length}(\text{lub}(a, b))+1} \leq b_{\text{length}(\text{lub}(a, b))+1}$$

なお $\text{pred}(a, b)$ は a, b のうち行きがけ順で先に表されるものを求める関数である。

【補正方法】変更

【補正内容】

【手続補正 2】

【0 1 2 1】同様に、述語 succ を次の様に定義す

【補正対象書類名】明細書

10 る。

【補正対象項目名】0 1 2 1

$$\text{succ}(a, b) \Leftrightarrow a_{\text{length}(\text{lub}(a, b))+1} \geq b_{\text{length}(\text{lub}(a, b))+1}$$

なお $\text{succ}(a, b)$ は a, b のうち行きがけ順で後に表されるものを求める関数である。

に表されるものを求める関数である。

フロントページの続き

(72) 発明者 村田 真

神奈川県横浜市保土ヶ谷区神戸町134番地

横浜ビジネスパーク イーストタワー

20

富士ゼロックス株式会社内